NASA
Technical Memorandum 107204

Army Research Laboratory
Memorandum Report ARL–MR–307

ALLSPD-3D
Version 1.0a

K.-H. Chen
Ohio Aerospace Institute
Brook Park, Ohio

B. Duncan
NYMA, Inc.
Brook Park, Ohio

D. Fricker
Vehicle Propulsion Directorate
U.S. Army Research Laboratory
Lewis Research Center
Cleveland, Ohio

J. Lee and A. Quealy
NYMA, Inc.
Brook Park, Ohio

April 1996

National Aeronautics and
Space Administration

U.S. ARMY
RESEARCH LABORATORY

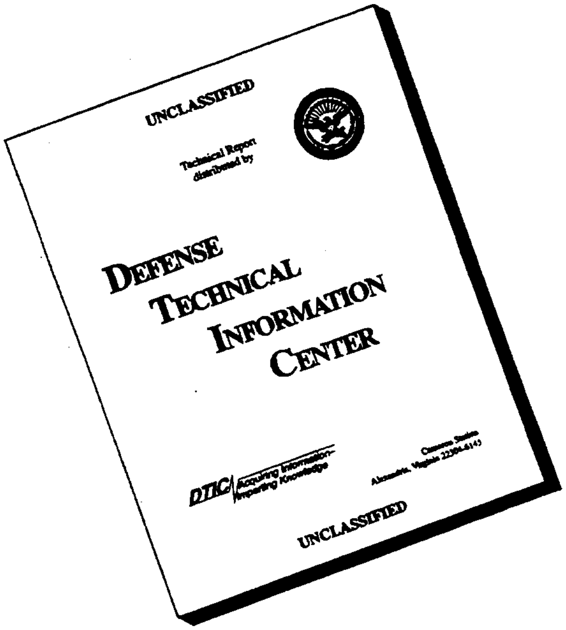
DISTRIBUTION POLICY

NASA policy, as outlined in NASA Management Instruction (NMI) 2210.2B, paragraph 6, subparagraph a(5), allows the release of preliminary or development versions of computer codes only if the recipient assists in the development process. Organizations receiving the code will, therefore, be expected to inform the Internal Fluid Mechanics Division of NASA Lewis Research Center of problems encountered in using the code and the documentation. We expect that you will run the example cases to confirm the successful installation of the code and to become familiar with the operation of the code.

Release of the code is restricted to domestic organizations whose major computational activity is conducted within the United States. In addition, the use of the code must be limited to the recipient, and further distribution of the code by the recipient is prohibited. Neither NASA Lewis Research Center, nor any of its employees, assumes any responsibility or legal liability for the accuracy or completeness of the code or represents that its use would not infringe on privately owned rights.

If the results of the code's use or incorporation with other code(s) are presented in oral or written form, please give credit for its development to the Internal Fluid Mechanics Division of NASA Lewis Research Center.

The code and documentation are still in development. Input from you on your experience with the code and its documentation will aid in their development. If you find errors, or have questions or comments, please contact the ALLSPD team via email at *allspd@lerc.nasa.gov* or contact David M. Fricker at 216-433-5960 or Kuo-Huey Chen at 216-962-3092.



Foreword

Aircraft engine companies in the U.S. have been using reacting flow CFD computer codes for many years as a design tool for new combustor designs and for the analysis of flow inside existing combustors to solve design problems. In spite of this record of successful use of CFD codes for combustors, the engine companies all agree that there is a need for combustor codes with capabilities much greater than those currently in use. Generally, current codes are limited in accuracy, are time-consuming in establishing a grid system which represents the complex geometries of a combustor, present difficulties in obtaining a stable solution (limited robustness), and require large execution times on mainframe supercomputers for practical problems. At the same time, the down-sizing that has occurred within these companies has left them with very limited resources to independently pursue an improved combustor code. They envision a partnership with the U.S. Government (NASA, Army, Air Force, etc.) to be a more cost-effective approach for producing improved combustor code capability that can be used by all within the U.S. without compromising proprietary, company-owned combustor designs. We at Lewis Research Center are pursuing this approach in the following way.

Efficient and accurate CFD algorithms have been developed for compressible flow applications over the last several years. These techniques have been modified for computational efficiency at the low Mach numbers which are typically found in combustors. Added to this numerical algorithm are state-of-the-art physical models for liquid fuel spray, complex turbulence, and finite rate chemistry for combustion. This Workshop marks the formal release of ALLSPD-3D to the U.S. combustion community. Concurrent with the ALLSPD effort, we are pursuing even more promising CFD approaches such as unstructured grids, finite element flow solvers, and adaptive re-meshing. The hallmark of these longer term efforts is the initiation of the National Combustor Code (NCC), a working partnership with several U.S. companies, NASA, and other agencies to jointly develop the next generation code. The first version of the NCC is expected to be available in 1998.

We believe that the ALLSPD-3D code being released at this time will be a major step forward in computational capability for its users. It should provide a user-friendly platform for executing the analyses of complex reacting flow problems with state-of-the-art physical models. Greater physical model fidelity and an improved gridding methodology are being pursued through the NCC partnership and will be available in the future. In the meantime, it is our desire that ALLSPD-3D provide computational capability to the U.S. industry until the fruits of the NCC ripen and are harvested.

The ALLSPD combustor code has been in development at Lewis since 1991. An initial 2-D version of the code was released in 1993. Some of the major features of the present 3-D code are a stochastic fuel spray model, a k-ε turbulence model, a library of finite rate combustion models, internal obstacles, multiple grid zone capability, and parallel processor execution. ALLSPD-3D has a graphical user interface to increase the comfort level of the user and has been run on supercomputers, a variety of scientific workstations, and a cluster of workstations. The code has successfully computed a wide range of reacting and non-reacting flow problems of increasing geometric complexity. The full validation of ALLPSD-3D will be an interactive process with the users as they gain experience with the code. The Lewis Research Center will work with the ALLSPD-3D users as they perform calculations. We are looking forward to a productive working relationship with you, the "customers," of our code development effort!

Dr. Edward J. Mularz
Chief, Engine Components Division
U.S. Army Research Laboratory - Vehicle Propulsion Directorate
NASA Lewis Research Center
November 14, 1995

(Dr. Mularz, as Chief of the Aerothermochemistry Branch at NASA Lewis Research Center, initiated the ALLSPD Code effort and guided its development until the recent move to his new position.)

Table of Contents

List of Figures

List of Tables

Abstract

This report describes the ALLSPD-3D computer program for the calculation of three-dimensional, chemically reacting flows with sprays. ALLSPD-3D employs a coupled, strongly implicit solution procedure for turbulent spray combustion flows. ALLSPD-3D utilizes generalized curvilinear coordinates with both multi-block and flexible internal blockage capabilities to handle complex geometries. The ALLSPD-3D algorithm, which employs preconditioning and eigenvalue rescaling techniques, is capable of providing efficient solution of flows over a wide range of subsonic Mach numbers. The chemistry treatment in the code is general enough that an arbitrary number of reactions and species can be defined by the users. In addition, for general industrial combustion applications, the code provides both dilution and transpiration cooling capabilities. A stochastic spray model and an efficient method for treating the spray source terms in the gas-phase equations are used to analyze evaporating spray droplets. Although written for three-dimensional flows, the code can be used for two-dimensional and axisymmetric flow computations as well. The code is written in such a way that it can be run on various computer platforms (supercomputers, workstations, and parallel processors) while the GUI (Graphical User Interface) should provide a user-friendly tool for setting up and running the code.

Nomenclature

Symbols

A,B,C	Convective Jacobians
A_0	Eddy Viscosity Modeling Parameter, Equation (40)
$A_{f,}$ A_b	Finite Reaction Rate Pre-exponents, Equation (25)
a_1, a_2, a_3	Turbulence Model Damping Coefficients, Equation (39)
C_j	Species Concentration
$C_{\varepsilon 1}$,$C_{\varepsilon 2}$	Turbulence Model Constants, Table 1
C_μ	Eddy Viscosity Constant, Equation (39)
CR_{ik}	Difference Between ν'_{ik} and ν''_{ik},Equation (26)
CS	Species Symbol, Equation (23)
D_{ij}	Diffusion Coefficients[m^2/s], Equation (44)
$E_{b,f}$	Activation Energy for Backward and Forward Reactions
G	Gibbs Free Energy
g	Specific Gibbs Free Energy
$\mathbf{H}$	Source Vector
H	Total Enthalpy $(h+(u^2+v^2+w^2)/2)$
h	Mixture Enthalpy [Joules/kg]
h_i	Specific Enthalpy of i^{th} species [Joules/kg]
$\bar{\boldsymbol{h}}$	Convective Heat Transfer Coefficient
$\mathbf{I}$	Identity Matrix
K	Finite Rate Reaction Constants
k	Turbulent Kinetic Energy
LHS	Left Hand Side
l	Length Scale [m]
MW_i	Molecular Weight of i^{th} Species

$\dot{m}$	Mass Flow Rate [kg/s]
$\dot{m}_p'''$	Evaporation Rate [kg/s/m^2]
N_r	Number of Chemical Reactions
N	Number of Chemical Species
p	Pressure, [N/m^2]
p_{atm}	$1.01325x10^5$ [N/m^2]
RHS	Right Hand Side
P_k	Turbulent Production, Equation (36)
Pr	Prandtl Number
r_p	Spray Particle Radius.
R_u	Universal Gas Constant
R_t	Turbulent Reynolds Number ($k^2/\nu\varepsilon$)
R_k	Turbulent Reynolds Number ($k^{1/2}y_n/\nu$)
Rp	p_{atm}/R_u
S_{ij}	Strain Rate Tensor, Equation (40)
T	Temperature
T_u	Turbulence Intensity
T_{ijk}	Source Jacobian
u,v,w	Mean Velocity Components
U_τ	Friction Velocity ($(\tau_w/\rho)^{1/2}$)
U_{ref}	Reference Velocity
x,y,z	Coordinates
X_i	Mole Fraction
y_n	Nearest Distance to a No-Slip Surface
y^+	Non-Dimensional Distance
Y_i	Species Mass Fraction

β	Kinetic energy per unit mass(u^2+v^2+w^3)
ε	Specific Dissipation
δ_{ij}	Kronecker Delta
Γ	Preconditioning Vector
κ	Thermal Conductivity, [Joules/(m-s-K)]
Λ	Low Reynolds Number Terms
μ	Viscosity [kg//m/s]
μ_t	Eddy Viscosity [kg/m/s]
ν	Kinematic Viscosity [m^2/s]
ν'_{ik} ,ν''_{ik}	Stoichiometric Coefficients for i^{th} Species in k^{th} Reaction
σ	Viscous Diffusion Coefficient, Table 1
ρ	Density
τ	Shear Stress
τ_{ij}	Reynolds Stresses, Equation (37)
ω_{ij}	Rate of Change of Species Concentration [kg of i^{th} species/(m^3-s)], Equation (24)

Subscripts and Superscripts

a, b, c	With Respect to the Directions
eff	Effective Quantities
inj	With Respect to Injector Holes, and Slots
k	With Respect to Turbulent Kinetic Energy
m	Group Index
p	With Respect to p^{th} Paticle Group
n	Pseudo Time Index
ref	With Respect to Reference Condition
t	With Respect to Turbulence
v	With Respect to Viscous Terms
^	Transformed Variable
∞	Freestream
0	At the Inlet

e	With Respect to Specific Dissipation
κ	With Respect to Conductivity
μ	With Respect to Viscosity

I : Introduction

It has been about two decades[1] since the U.S. Government sponsored a major Computational Fluid Dynamics (CFD) code development effort focused on the analysis of reacting flows inside combustors of gas turbine engines or liquid rocket engines. During the ensuing years, many advances in CFD have been realized, especially in the area of compressible flows. Applying these advanced techniques in a new reacting flow code has the potential for significant improvements in accuracy, efficiency, and robustness over the current design codes being used in the industry. For the past few years, NASA Lewis Research Center has undertaken the development of a new combustor analysis code, ALLSPD-3D, which employs state-of-the-art computational technology and physical models. During this development, NASA Lewis will provide from time to time a current version of the code to the aerospace industry, and will solicit feedback from users on the code's shortcomings and suggestions for further improvements. (Please direct any questions or comments regarding the ALLSPD-3D code to *allspd@lerc.nasa.gov.*)

The current ALLSPD-3D code has evolved from the two-dimensional version released in June, 1993. Besides extension to three dimensions, the new code features several additional improvements, including a user-friendly graphical user interface (GUI), multi-platform capability (supercomputers, workstations, and parallel processors), improved turbulence and spray models, generalized chemistry, a new solution procedure, and more general boundary conditions. In this report, the fundamental structure of the ALLSPD-3D code, numerics, physical models, and I/O procedure are described in detail.

The numerics of the ALLSPD-3D code are based on modern compressible algorithms to take advantage of advances made in CFD during the past two decades. It is well known that numerical methods developed for compressible flows are often ineffective at low Mach numbers. There are two reasons[2,6] for this difficulty. First, the system's eigenvalues become stiff at low flow velocities. In theory, this can be circumvented by using implicit numerical schemes. In practice, however, a large approximate factorization error in multiple dimensions limits an optimal CFL number beyond which convergence slows down. At low Mach numbers, there are large disparities among CFL numbers and eigenvalues in three directions leads to further uncertainties. Therefore, the optimal CFL numbers can be difficult to determine. Second, the pressure term in the momentum equations becomes singular as the Mach number approaches zero, yielding a large round-off error. This behavior smears the pressure variation and often produces inaccurate solutions. Following the approach in Shuen et. al.[6] and Chen,[7] these difficulties are circumvented in the ALLSPD-3D code by using the pressure decomposition and the eigenvalue rescaling techniques. The results of these treatments show that the convergence characteristics of the code can indeed formulated to be independent of the flow Mach number.[6]

Chemically reacting flows exhibit another class of numerical difficulties because of the wide range of time scales involved in the calculation. To avoid this chemical stiffness problem, the chemical source terms are treated fully implicitly. This measure is equivalent to preconditioning the time derivative terms of the species conservation equations so that all chemical and convective processes proceed at approximately the same numerical rate.

As pointed out in Shuen et. al.,[6] the Navier-Stokes computation of a general three-dimensional turbulent combustion flow has been a tremendous task in the past and even with today's supercomputers and high end workstation clusters, it is still very difficult to accomplish such a demanding job. To understand the magnitude of such computations, it is useful to analyze a typical three-dimensional reacting flow problem with detailed chemistry. The compressible Navier-Stokes equations are usually required because of the wide density variations resulting from the combustion process despite the low flow speed in most practical applications. In addition, the turbulent flowfield must be modeled for most cases with only few exceptions. Thus, with the standard k-ε equations to model the turbulence, seven equations are required to solve for turbulent non-reacting flows. For reacting flows with N species considered, the total number of equations to be solved for a three-dimensional turbulent reacting flow is $7+(N-1)$, and the number of species (N) to completely describe a typical hydrocarbon fuel reaction is generally greater than 40.[10] Coupling this number of equations to be solved with a grid suitable to describe a typical combustor geometry (250,000+ cells) quickly exceeds the capabilities of existing computer technology. It is definitely impractical to include so many species and couple with the flow equations in terms of today's (and in the near future's) computer technology. In the past, ways have been sought to reduce the number of species (reduced mechanism) for some particular fuels in order to form a more tractable CFD problem.[9] Unfortunately, although significantly reduced, the number of species is still between 10 and 20 depending on the particular application. Further reduction below this level is unlikely because the reduction will also lead to inadequate description of the combustion physics.[10]

The numerical issues relating to solving the governing equations for turbulent reacting flows are robustness, CPU time requirements, implicitness, stability, equation coupling, and memory. Some of these issues are very closely related, such as robustness, implicitness, stability and equation coupling. During the past two decades, the development of numerical algorithms and the usage of various solvers in CFD have been more or less dependent on the availability and accessibility of computer resources. The trend has evolved from explicit, decoupled schemes to strongly implicit and fully coupled strategies. The pros and cons among various schemes depend on the problems of interest and sometimes the vectorization and/or parallelization on computer platforms as well. Nevertheless, strongly implicit, fully coupled schemes generally give better stability, robustness and convergence, at least for steady state calculations; however, they usually take more CPU time per iteration compared to explicit schemes. For combustion flow computations, equation coupling (at least for species equations) is inevitable due to the strong coupling nature of the chemical source terms in a detailed chemistry treatment.

In order to compromise between the degree of coupling and the total memory requirement, the partially decoupled numerical scheme in developed by Chen and Shuen[11] has been adopted for ALLSPD-3D. This procedure divides the whole set of equations into three groups. The first group is the five flow equations, the second group is the two turbulent equations and the third group is the species equations. Each group of equations is solved in a coupled manner and the subsequent sweep through the three groups completes one iteration. This arrangement provides greater flexibility in various types of flow calculations while maintaining the strongly coupled and implicit manner within the existing combustion code structure. The numerical issues relating to this partially decoupled process are described in Chen and Shuen[11].

The solver in the ALLSPD-3D code has been modified to include the LU scheme which, through our tests,[7] is thought to be more suitable for solving large sets of coupled three-dimensional reacting flow equations.

A recently developed low Reynolds number k-ε model[12] is used to represent the effects of turbulence. In this model, the k-ε equations are solved all the way to the wall and the conventional wall function treatment of near wall regions is not needed. Some variations of implementing this model, especially the near wall treatment, are also available as an option to alleviate the fine grid requirements near walls (which is typically a heavy burden associated with this type of turbulence model for three-dimensional calculations). Detailed treatment of the turbulent model is discussed later in this report. Although turbulence is included, the mean flow quantities are still used in the chemistry calculations (laminar chemistry) in the current version of the code. A simple eddy breakup model (EBM)[13] is available as an option to model the turbulence effect on combustion.

The liquid phase is treated by solving Lagrangian equations of motion and transport for the life histories of a statistically significant sample of individual droplets. This involves dividing the droplets into n groups (defined by position, velocity, temperature, and diameter) at the fuel nozzle exit and then computing their subsequent trajectories in the flow. The spray model used in this study is based on a dilute spray assumption which is valid in regions where the droplet loading is low.[14] The liquid fuel is assumed to enter the combustor as a fully atomized spray comprised of spherical droplets. The present model does not account for the effects due to droplet breakup and coalescence processes which might be significant in a dense spray situation. A stochastic process is included in the spray calculation to account for the effect of turbulence dispersion on the spray trajectories. Also the temperature within each droplet is non-uniform and a vortex model[15] is used to compute the internal temperature distribution.

Most combustors in aero-propulsive engines involve very complex geometries. To enhance geometrical flexibility, the ALLSPD-3D code is written in generalized curvilinear coordinates and is equipped with multi-block grid and internal blockage capabilities. These features allow the users to handle complex geometries. Some of the main features of ALLSPD-3D are listed in Table 1

TABLE 1. ALLSPD-3D Main Features

• 3D/2D/ axisymmetric geometries
• Finite-difference, compressible flow formulation with low Mach number preconditioning (current version is intended only for subsonic flow simulations and uses central-differencing for convective and viscous terms on right and left-hand sides)
• Strongly coupled, implicit steady-state algorithm
• Generalized finite rate chemistry
• Generalized property and chemical reactions database
• Eddy breakup model for turbulence-chemistry closure
• Low-Reynolds number k-ε turbulence model
• Spray combustion (a stochastic, separated flow model with vortex model for droplet internal temperature distribution)
• Dilution and transpiration cooling capabilities
• Serial & parallel versions (using PVM for message passing language) on various platforms (IBM, SGI, SUN, HP workstations, CRAY)
• GUI for problem setup and post-processing
• Various boundary conditions (multiple inlets/outlets, dilution holes, transpiration holes, periodic, symmetry, far-field, adiabatic or isothermal walls, centerline singularity)

II : Mathematical Formulations

2.1 Gas-Phase Equations

The three-dimensional, unsteady, compressible, density-weighted, time-averaged Navier-Stokes equations and species transport equations for a chemically reacting gas of *N* species written in generalized non-orthogonal coordinates can be expressed as

$$\frac{\partial}{\partial\tau}\tilde{Q}+\frac{\partial}{\partial\xi}\left(\tilde{E}-\tilde{E}_{\nu}\right)+\frac{\partial}{\partial\eta}\left(\tilde{F}-\tilde{F}_{\nu}\right)+\frac{\partial}{\partial\zeta}\left(\tilde{G}-\tilde{G}_{\nu}\right) = \tilde{H}_c+\tilde{H}_l \qquad (1)$$

where the vectors $\tilde{Q}$, $\tilde{E},\tilde{F},\tilde{G},\tilde{E}_{\nu},\tilde{F}_{\nu},\tilde{G}_{\nu},\tilde{H}_c$ and $\tilde{H}_l$ are defined as

$$\begin{array}{ll}
\multicolumn{2}{c}{\tilde{Q}=\frac{1}{J}Q} \\
\tilde{E}=\frac{1}{J}(\xi_t Q+\xi_x E+\xi_y F+\xi_z G) & \tilde{E}_{\nu}=\frac{1}{J}(\xi_x E+\xi_y F+\xi_z G) \\
\tilde{F}=\frac{1}{J}(\eta_t Q+\eta_x E+\eta_y F+\eta_z G) & \tilde{F}_{\nu}=\frac{1}{J}(\eta_x E+\eta_y F+\eta_z G) \\
\tilde{G}=\frac{1}{J}(\varsigma_t Q+\varsigma_x E+\varsigma_y F+\varsigma_z G) & \tilde{G}_{\nu}=\frac{1}{J}(\varsigma_x E+\varsigma_y F+\varsigma_z G) \\
\tilde{H}_c=\frac{1}{J}H_c & \tilde{H}_l=\frac{1}{J}H_l
\end{array} \qquad (2)$$

In the above expressions, τ, ξ, η and ζ are the time and generalized spatial coordinates, and ξ_t, η_t and ζ_t are the grid speed terms. The variables ξ_x, ξ_y, ξ_z, η_x, η_y, η_z, ζ_x, ζ_y and ζ_z are the metric terms and *J* is the transformation Jacobian. The vectors $\boldsymbol{Q}, \boldsymbol{E}, \boldsymbol{F}, \boldsymbol{G}, \boldsymbol{E}_v, \boldsymbol{F}_v, \boldsymbol{G}_v, \boldsymbol{H}_c$ and $\boldsymbol{H}_l$ in the above definitions are

$$
\begin{aligned}
Q &= \left(\rho,\ \rho u,\ \rho v,\ \rho w,\ \rho E_t,\ \rho k,\ \rho\varepsilon,\ \rho Y_1 \ldots \rho Y_{N-1}\right)^T \\
E &= \left(\rho u,\ \rho u^2+p,\ \rho uv,\ \rho uw,\ \left(\rho E_t+p\right)u,\ \rho ku,\ \rho\varepsilon u,\ \rho u Y_1 \ldots \rho u Y_{N-1}\right)^T \\
F &= \left(\rho v,\ \rho uv,\ \rho v^2+p,\ \rho vw,\ \left(\rho E_t+p\right)v,\ \rho kv,\ \rho\varepsilon v,\ \rho v Y_1 \ldots \rho v Y_{N-1}\right)^T \\
G &= \left(\rho w,\ \rho uw,\ \rho vw,\ \rho w^2+p,\ \left(\rho E_t+p\right)w,\ \rho kw,\ \rho\varepsilon w,\ \rho w Y_1 \ldots \rho w Y_{N-1}\right)^T \\
E_v &= \left(0, \tau_{xx}, \tau_{xy}, \tau_{xz}, u\tau_{xx}+v\tau_{xy}+w\tau_{xz}+q_x, \tau_{kx}, \tau_{\varepsilon x}, q_{x1} \ldots q_{xN-1}\right)^T \\
F_v &= \left(0, \tau_{yx}, \tau_{yy}, \tau_{yz}, u\tau_{yx}+v\tau_{yy}+w\tau_{yz}+q_y, \tau_{ky}, \tau_{\varepsilon y}, q_{y1} \ldots q_{yN-1}\right)^T \\
G_v &= \left(0, \tau_{zx}, \tau_{zy}, \tau_{zz}, u\tau_{zx}+v\tau_{zy}+w\tau_{zz}+q_z, \tau_{kz}, \tau_{\varepsilon z}, q_{z1} \ldots q_{zN-1}\right)^T
\end{aligned}
\tag{3}
$$

and the source term vectors H_c and H_l are

$$
H_c = \left(0, 0, 0, 0, 0, 0, S_k, S_\varepsilon + \Lambda, \omega_1 \ldots \omega_{N-1}\right)^T \tag{4}
$$

$$
\begin{aligned}
&H_l = \\
&\Bigg(\sum_p n_p \dot{m}_p,\ \sum_p n_p \dot{m}_p u_p - \frac{4\pi}{3}\rho_p r_p^3\ n_p \frac{du_p}{dt},\ \sum_p n_p \dot{m}_p v_p - \frac{4\pi}{3}\rho_p r_p^3\ n_p \frac{dv_p}{dt}, \\
&\sum_p n_p \dot{m}_p w_p - \frac{4\pi}{3}\rho_p r_p^3\ n_p \frac{dw_p}{dt},\ \sum_p n_p \dot{m}_p h_{fs} - 4\pi r_p^2\ n_p \bar{h}\Delta T, 0, 0, \\
&\sum_p n_p \dot{m}_p, 0, 0, \ldots\ldots, 0, 0\Bigg)^{\mathrm{T}}
\end{aligned}
\tag{5}
$$

where ρ, p, u, v, w, κ, ε and Y_i represent the density, pressure, Cartesian velocity components, turbulent kinetic energy, dissipation rate of turbulent kinetic energy and species mass fraction, respectively; E_t=e+1 /2(u^2+v^2+w^2) is the total internal energy with e being the thermodynamic internal energy; and ω_i is the rate of change of species i due to chemical reactions. The ω_i in the chemical source term will be described later in Section 2.4. The quantities related to the source term in the turbulent equations are described in Section 2.6. The normal and shear stresses, energy, species, and turbulent diffusion fluxes are given by

$$
\begin{array}{ll}
q_{xi} = \rho D_{im}\dfrac{\partial Y_i}{\partial x} & \tau_{xx} = 2\mu_e\dfrac{\partial u}{\partial z} - \dfrac{2}{3}\mu_e\left(\dfrac{\partial u}{\partial x} + \dfrac{\partial v}{\partial y} + \dfrac{\partial w}{\partial z}\right) \\
q_{yi} = \rho D_{im}\dfrac{\partial Y_i}{\partial y} & \tau_{yy} = 2\mu_e\dfrac{\partial v}{\partial y} - \dfrac{2}{3}\mu_e\left(\dfrac{\partial u}{\partial x} + \dfrac{\partial v}{\partial y} + \dfrac{\partial w}{\partial z}\right) \\
q_{zi} = \rho D_{im}\dfrac{\partial Y_i}{\partial z} & \tau_{zz} = 2\mu_e\dfrac{\partial w}{\partial z} - \dfrac{2}{3}\mu_e\left(\dfrac{\partial u}{\partial x} + \dfrac{\partial v}{\partial y} + \dfrac{\partial w}{\partial z}\right) \\
\tau_{xk} = \left(\mu + \dfrac{\mu_t}{\sigma_k}\right)\dfrac{\partial k}{\partial x} & \tau_{xy} = \tau_{yx} = \mu_e\left(\dfrac{\partial u}{\partial y} - \dfrac{\partial v}{\partial x}\right) \\
\tau_{yk} = \left(\mu + \dfrac{\mu_t}{\sigma_k}\right)\dfrac{\partial k}{\partial y} & \tau_{xz} = \tau_{zx} = \mu_e\left(\dfrac{\partial u}{\partial z} - \dfrac{\partial w}{\partial x}\right) \\
\tau_{zk} = \left(\mu + \dfrac{\mu_t}{\sigma_k}\right)\dfrac{\partial k}{\partial z} & \tau_{yz} = \tau_{zy} = \mu_e\left(\dfrac{\partial v}{\partial z} - \dfrac{\partial w}{\partial y}\right) \\
\tau_{x\varepsilon} = \left(\mu + \dfrac{\mu_t}{\sigma_\varepsilon}\right)\dfrac{\partial \varepsilon}{\partial x} & q_{xe} = k_e\left(\dfrac{\partial T}{\partial x}\right) + \rho\displaystyle\sum_{i=1}^{N} h_i D_{im}\dfrac{\partial Y_i}{\partial x} \\
\tau_{y\varepsilon} = \left(\mu + \dfrac{\mu_t}{\sigma_\varepsilon}\right)\dfrac{\partial \varepsilon}{\partial y} & q_{ye} = k_e\left(\dfrac{\partial T}{\partial y}\right) + \rho\displaystyle\sum_{i=1}^{N} h_i D_{im}\dfrac{\partial Y_i}{\partial y} \\
\tau_{z\varepsilon} = \left(\mu + \dfrac{\mu_t}{\sigma_\varepsilon}\right)\dfrac{\partial \varepsilon}{\partial z} & q_{ze} = k_e\left(\dfrac{\partial T}{\partial z}\right) + \rho\displaystyle\sum_{i=1}^{N} h_i D_{im}\dfrac{\partial Y_i}{\partial z}
\end{array}
\qquad (6)
$$

where T, μ_e, μ_t, μ and k_e are the temperature, effective viscosity, turbulent viscosity, molecular viscosity and the effective thermal conductivity, respectively. The vector H_l in Equation (1) represents the source term that accounts for the interactions between the gas and liquid phases. The n_p is the number of droplets in the p^{th} characteristic group of droplets. The $\dot{m}_p$ is the evaporation rate of each particle group; ρ_p is the liquid density. The r_p is the droplet radius. The u_p, v_p and w_p are the particle velocities for the p^{th} particle group. The h_{fs} is the enthalpy of fuel vapor at the droplet surface, and $\overline{h}\,\Delta\,T$ is the convective heat transfer between two phases. Detailed discussions about the liquid phase equations will be described in Section 2.3. In reacting flow calculations, the evaluation of thermo-physical properties is of vital importance. In this code, the values of C_{pi}, k_i, and μ_i for each species are determined by fourth-order polynomials in temperature, as described in Section 2.7 and Section 2.8.

2.2 All-Mach-Number Formulation

As noted earlier, the two main difficulties that render the compressible flow algorithms ineffective at low Mach numbers are the round-off error caused by the singular pressure gradient term in the momentum equations (the pressure term is of order $1/M^2$ while the convective term is of order unity in the non-dimensional momentum equations) and the stiffness caused by the wide disparities in eigenvalues. To circumvent these two problems regarding the low Mach number calculations, a time preconditioning term has been applied to rescale the system eigenvalues and the pressure variable is decomposed into a constant reference pressure (p_{ref}) and a gauge pressure (p_g). This approach has been successfully applied to solve chemical non-equilibrium viscous flows with and without sprays by Shuen et. al.[6] and by Chen and Shuen.[7] After including the all-Mach-number formulation, the resulting Navier-Stokes equations in a conservative form are

$$\Gamma\frac{\partial}{\partial\tau^*}(\hat{Q}) + \frac{\partial}{\partial\tau}\tilde{Q} + \frac{\partial}{\partial\xi}\left(\tilde{E} - \tilde{E}_v\right) + \frac{\partial}{\partial\eta}\left(\tilde{F} - \tilde{F}_v\right) + \frac{\partial}{\partial\zeta}\left(\tilde{G} - \tilde{G}_v\right) = \tilde{H}_c + \tilde{H}_l \qquad (7)$$

where the primitive variable vector $\hat{Q}$ and the preconditioned matrix are given as

$$\hat{Q} = \left(p_g, u, v, w, h, k, \varepsilon, Y_1 \ldots Y_{N-1}\right)^T \qquad (8)$$

$$\Gamma = \begin{bmatrix} \frac{1}{\beta} & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ \frac{u}{\beta} & \rho & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ \frac{v}{\beta} & 0 & \rho & 0 & 0 & 0 & 0 & \cdots & 0 \\ \frac{w}{\beta} & 0 & 0 & \rho & 0 & 0 & 0 & \cdots & 0 \\ \frac{H}{\beta} - 1 & \rho u & \rho v & \rho w & \rho & 0 & 0 & \cdots & 0 \\ \frac{k}{\beta} & 0 & 0 & 0 & . & \rho & \cdots & 0 & 0 \\ \frac{\varepsilon}{\beta} & 0 & 0 & 0 & . & . & \rho & \cdots & 0 \; 0 \\ \frac{Y_1}{\beta} & 0 & 0 & 0 & 0 & . & . & \rho \; . \; . \; . & 0 \\ \frac{Y_2}{\beta} & 0 & 0 & 0 & 0 & 0 & . & \vdots \; \rho \; . \; . & 0 \\ . & . & . & . & . & . & 0 & . \; . \; \rho \; . & 0 \\ . & . & . & . & . & . & . & 0 \; . \; . \; \rho & 0 \\ \frac{Y_{N-1}}{\beta} & 0 & 0 & 0 & 0 & 0 & . & . \; 0 \; . \; . & \rho \end{bmatrix} \tag{9}$$

where τ^* is the pseudo-time, $h=e+p/\rho$ is the specific enthalpy of the gas mixture. The eigenvalues (in the ξ-direction) in pseudo-time can be obtained from the matrix $\Gamma^{-1}A$, where A is the Jacobian $\partial(\tilde{E})/\partial\hat{Q}$. The eigenvalues for a real gas are

$$\lambda = \left(U, U, U, \ldots, \frac{1}{2}\left[U\left(1 + \frac{\beta}{c^2}\right) \pm \sqrt{U^2\left(1 - \frac{\beta}{c^2}\right) + 4\beta\left(\alpha^2{}_1 + \alpha^2{}_2 + \alpha^2{}_3\right)} \right] \right) \tag{10}$$

where c is the speed of sound, $\alpha_1 = \xi_x$, $\alpha_2 = \xi_y$, $\alpha_3 = \xi_z$, and U is the contravariant velocity component defined as $U = \alpha_1\, u + \alpha_2\, v + \alpha_3\, w$. To obtain well-conditioned eigenvalues, the scaling factor β is taken to be $\beta = u^2+v^2+w^2$; such that all the eigenvalues will have the same order of magnitude. The definition of vectors in Equation (7) is identical to those in Equation (1) except that the pressure terms in the momentum equations are replaced by the gauge pressure.

2.3 Liquid-Phase Equations

2.3.1 Droplet Motion Equations

The liquid phase is treated by solving Lagrangian equations of motion and transport for the life histories of a statistically significant sample of individual droplets. This involves dividing the droplets into *n* groups (defined by position, velocity, temperature and diameter) at the fuel nozzle exit and then computing their subsequent trajectories in the flow. The spray model used in this study is based on a dilute spray assumption which is valid in regions where the droplet loading is low. The liquid fuel is assumed to enter the combustor as a fully atomized spray comprised of spherical droplets. The present model does not account for the effects due to droplet breakup and coalescence processes, which might be significant in a dense spray situation. The Lagrangian equations governing the droplet motion are

$$\begin{aligned}
\frac{dx_p}{dt} &= u_p & \qquad & & \frac{du_p}{dt} &= \frac{3}{16}\frac{C_D\mu_g Re_p}{\rho_p r_p^2}(u_g - u_p) \\
\frac{dy_p}{dt} &= v_p & \qquad & \text{and} & \frac{dv_p}{dt} &= \frac{3}{16}\frac{C_D\mu_g Re_p}{\rho_p r_p^2}(v_g - v_p) \qquad (11) \\
\frac{dz_p}{dt} &= w_p & \qquad & & \frac{dw_p}{dt} &= \frac{3}{16}\frac{C_D\mu_g Re_p}{\rho_p r_p^2}(w_g - w_p)
\end{aligned}$$

where the particle Reynolds number, Re_p, and the drag coefficient, C_D, are defined as

$$Re_p = \frac{2r_p\rho_g}{\mu_g}\left[(u_g - u_p)^2 + (v_g - v_p)^2 + (w_g - w_p)^2\right]$$

$$C_D = \begin{cases} \dfrac{24}{Re_p}\left(1 + \dfrac{Re_p^{2/3}}{6}\right) & \text{for} \quad Re_p < 1000 \\ 0.44 & \text{for} \quad Re_p \geq 1000 \end{cases} \qquad (12)$$

The subscript g represents the gas-phase quantities and p represents the liquid-phase (or "particle") quantities. Equation (11) and Equation (12) are used to calculate the new droplet positions, and the new droplet velocities at the new locations. A second-order Runge-Kutta scheme was used to integrate Equation (11).

2.3.2 Droplet Mass and Heat Transfer Equations

As described by Faeth,[14] the following correlations were used to approximate the mass and heat transfer coefficients for a single isolated spherical droplet,

$$\frac{\dot{m}''_p r_p}{\rho D_f} = N_s(1 + B) \qquad (13)$$

$$\frac{\bar{h} r_p}{k} = \frac{N_p \ln(1 + B)^{Le-1}}{[(1 + B)^{Le-1} - 1]} \qquad (14)$$

where $\bar{h}$ is the heat transfer coefficient, d_p is the droplet diameter, D_f is the fuel mass diffusivity, k is the thermal conductivity of fuel vapor and $\dot{m}''_p$ is the fuel mass evaporation rate per unit area. The N_s and N_p are defined as,

$$N_s = 1 + \frac{0.276 Re_p^{1/2} Pr^{1/3}}{\left[1 + \dfrac{1.232}{Re_p Pr^{3/4}}\right]^{1/2}} \quad \text{and} \quad N_p = 1 + \frac{0.276 Re_p^{1/2} Sc^{1/3}}{\left[1 + \dfrac{1.232}{Re_p Sc^{3/4}}\right]^{1/2}} \qquad (15)$$

where *Sc* and *Le* are the Schmidt and Lewis numbers, respectively. The Spalding number, *B*, in Equation (13) and Equation (14) is defined as,

$$B = \frac{Y_{fgp} - \overline{Y}_{fg}}{1 - Y_{fgp}} \tag{16}$$

where Y_{fgp} is the fuel vapor mass fraction at the surface of the droplet and $\overline{Y}_{fg}$ is the mean fuel mass fraction of the ambient gas. In the present study, Y_{fgp} is obtained from the following equation,

$$Y_{fgp} = \frac{X_{fgp} W_f}{X_{fgp} W_f + (1 + X_{fgp}) W_a} \tag{17}$$

where W_a is the molecular weight of gas mixture excluding fuel vapor, W_f is the molecular weight of fuel and X_{fgp} is the mole fraction of fuel. The X_{fgp} is obtained from the assumption of Raoult's law. Based on this assumption, the mole fraction at the droplet surface is equal to the ratio of the partial pressure of fuel vapor to the total pressure. For the present spray calculation, the partial pressure of fuel vapor is computed using the empirical correlation in Reid.[17]

$$\ln\left(\frac{p_{vp}}{p_c}\right) = \frac{1}{1-x}\left(P_{v1}x + P_{v2}x^{1.5} + P_{v3}x^{3} + P_{v4}x^{6}\right) \tag{18}$$

where x=1-T/T_c, T_c and p_c are the critical temperature and critical pressure of the fuel vapor, respectively. The constants, P_{v1}, P_{v2}, P_{v3}, and P_{v4} are obtained from Reid.[17]

2.3.3 Droplet Internal Temperature Equations

As a single droplet enters a hot environment, the small portion of the droplet near the surface will heat up quickly while the center core of the droplet remains "cold." The heat will be conducted and convected to the droplet's interior as the droplet penetrates further into the hot ambient gas. Eventually, the temperature within the droplet will become nearly uniform before the end of its lifetime. To solve this transient process occurring within the droplet is not trivial. In the past, certain approximations were usually made to alleviate this computational burden while retaining the solution integrity. The simplest approximation is the uniform temperature model. This model assumes that the thermal conductivity of the fuel is infinite. Of course, this is not valid at the beginning of the heating-up process of the droplet. Another model considers the heat diffusion inside the droplet, yielding a conduction model. The temperature distribution within the droplet is obtained by solving the one-dimensional heat conduction equation subject to the convective boundary conditions at the droplet surface. The conduction model completely neglects the convective phenomena within the droplet which might occur due to significant shear forces at the surface induced by high slip velocities. Tong and Sirignano [15] developed a *vortex model* for the internal temperature of a single droplet which accounts for the convective effect of the Hill's vortex formed inside the droplet. Tong and Sirignano's vortex model is used in ALLSPD-3D code to obtain the internal temperature distribution of the droplet. The equations governing the internal temperature distribution based on this model are,

$$\frac{\partial T_p}{\partial t} = \frac{17k_l}{C_{pl}\rho_l r_p^2}\left[\alpha\frac{\partial^2 T_p}{\partial \alpha^2} + (1 + C(t)\alpha)\frac{\partial T_p}{\partial \alpha}\right] \quad \text{and} \quad C(t) = \frac{3}{17}\left(\frac{C_{pl}\rho_l}{k_l}\right)r_p\frac{dr_p}{dt} \tag{19}$$

where the value of α varies between 0 and 1 with α=0 referring to the vortex center and α=1 referring to the droplet surface. The initial and boundary conditions for Equation (19) are

$$\begin{array}{ll} t = t_{inj} & T_p = T_{inj} \\ \alpha = 0 & \dfrac{\partial T_p}{\partial \alpha} = \dfrac{1}{17}\left(\dfrac{C_{pl}\rho_l}{k_l}\right)\left(r_p^2\dfrac{\partial T_p}{\partial t}\right) \\ \alpha = 1 & \dfrac{\partial T_p}{\partial \alpha} = \dfrac{3}{16}\left(r_p\dfrac{\partial T_p}{\partial t}\right)_s \end{array} \tag{20}$$

where $\partial T_p / \partial \alpha$ is obtained from the energy balance at the droplet surface,

$$\left.\frac{\partial T_p}{\partial r}\right)_s = \frac{1}{k_l}[\bar{h}\Delta(T - \dot{m}_p h_{fg})] \qquad (21)$$

where $\dot{m}''_p$ and $\bar{h}$ are calculated from Equation (13) and Equation (14), respectively, h_{fg} is the latent heat of the fuel and $\Delta T = T_g\text{-}T_{ps}$, where T_{ps} is droplet surface temperature and $\tilde{T}_g$ is the mean gas temperature evaluated as,

$$\tilde{T}_g = \frac{1}{3}T_g + \frac{2}{3}T_{ps} \qquad (22)$$

An implicit scheme is used to solve Equation (19) subject to the initial and boundary conditions. Second-order central difference is used for the spatial differential terms and a first-order difference is used for the time term. These treatments render a scalar tridiagonal algebraic system and is solved by the Thomas algorithm.[18]

2.4 Chemical Source Term

For a set of N_r elementary reactions involving N species, the reaction equations can be written in the following general form,

$$\sum_{i=1}^{N} \mathrm{v}'_{ik} CS_i \leftrightarrows \sum_{i=1}^{N} \mathrm{v}''_{ik} CS_i \qquad , \mathrm{k} = 1,2,3 \ldots \mathrm{Nr} \qquad (23)$$

where v'_{ik} and v''_{ik} are the stoichiometric coefficients for species i appearing as a reactant in the k^{th} forward and backward reactions, respectively. The law of mass action states that the rate of change of concentration of the i^{th} species is defined by the movement of each species summed over all of the contributing reactions. Therefore, the rate of change in species concentration is expressed as follows,

$$\omega_i = MW_i \sum_{k=1}^{N_r} \left(\mathrm{CR}_{ik} \left(K_f(k) \prod_{j=1}^{N} C_j^{\nu'_{jk}} - K_b(k) \prod_{j=1}^{N} C_j^{\nu''_{jk}} \right) \right) \qquad (24)$$

The forward and backward reaction rate constants (K_f, K_b) are computed from Arrhenius' relationship with phenomenologically developed forms. The forward and backward reaction rate constants of the k^{th} reactions are defined as,

$$\begin{aligned} K_b(k) &= A_b(k)\, T^{\alpha_b(k)}\, exp\left(\frac{E_b(k)}{R_u T}\right) \\ K_f(k) &= A_f(k)\, T^{\alpha_f(k)}\, exp\left(\frac{E_f(k)}{R_u T}\right) \end{aligned} \qquad (25)$$

$$\mathrm{CR}_{ik} = \left(\nu''_{ik} - \nu'_{ik}\right) \qquad (26)$$

The influence of other bodies on a particular chemical reaction is modeled using a typical third body efficiency correction,

$$\begin{aligned} K_f(k) &= K_f(k) \sum_{i=1}^{N} \mathrm{Third}_i(k)\, C_i \\ K_b(k) &= K_b(k) \sum_{i=1}^{N} \mathrm{Third}_i(k)\, C_i \end{aligned} \qquad (27)$$

It is noted that, because of the vastly different time scales that may be involved in the elementary reactions and the flow, and the strong dependence of the source terms on temperature (exponentially) and density (ρ^2 or ρ^3, depending on the order of reaction), the set of equations may become very stiff for most flow conditions of interest. To mitigate this problem, chemical source terms are treated in an implicit fashion in the ALLSPD-3D code to improve numerical stability. Details for the construction of the source term Jacobian are described in Appendix A.

2.5 Equilibrium Rate Kinetics Model

There is a large database of finite-rate chemistry data where only one of the two required reaction rate constants is reported. The chemical rate information in the secondary direction, either forward or backward direction, can be determined from an equilibrium assumption. Typically, the equilibrium constant (K_p) for a reaction can be found in many standard chemistry references as a function of temperature. However, it is convenient to determine K_p as a part of the solution. This can be done by taking advantage of the large thermodynamic information available in the NASA Lewis database.[16] The equilibrium rate information can be determined for a large number of hydrocarbon related species using the following procedure.

In an equilibrium state, it is known that the partial pressure ratio of the species component of an ideal mixture is equal to the equilibrium constant (K_p).[34] In equilibrium $dG)_{T,p}$ is zero and therefore only the Gibbs free energy of formation for each species is needed to determine the equilibrium constant. For a simple reaction, a CS(a)+ b CS(b) $\leftrightarrows$ c CS(c)+d CS(d), the equilibrium constant in terms of partial pressures (nondimensionalized by atmospheric pressure) is defined as,

$$K_p = \frac{[P_a]^a [P_b]^b}{[P_c]^c [P_d]^d} = \exp\left(\frac{-\Delta G^0}{R_u T}\right) \tag{28}$$

where ΔG^0 is defined as standard free energy change. This relationship can be extended to any general chemical reaction using the standard free energy change expressed in terms of molar Gibbs free energy [kcal/kg-kmole]. The molar Gibbs free energy is defined by the following expression,

$$\Delta_f G_r^0 = \sum_{n=1}^{N} \nu''_{nr} \Delta g_{nr}^0 - \sum_{n=1}^{N} \nu'_{nr} \Delta g_{nr}^0 \qquad (29)$$

where the partial molar Gibbs free energy of formulation of n^{th} species, Δg_{nr}^0 , consistent with the polynomial representations of Gordon and McBride is,

$$\Delta g_n^0(T) = h - Ts = C_{p_n}(0)\,(1 - ln(T))\,T - C_{p_n}(1)\frac{T^2}{2} - C_{p_n}(2)\frac{T^3}{6}$$

$$- C_{p_n}(3)\frac{T^4}{12} - C_{p_n}(4)\frac{T^5}{20} + C_{p_n}(5) - C_{p_n}(6)\,T \qquad (30)$$

$$\Delta_f G^0{}_r(T) = \sum_{n=1}^{N} \mathrm{CR}_{nr} \Delta g_n^0(T) \qquad (31)$$

$$K_{P_r} = exp\left(\frac{-\Delta_f G^0{}_r(T)}{R_u T}\right) = \prod_{i=1}^{N} \frac{p_i^{\nu''_{ir}}}{p_i^{\nu'_{ir}}}\left(\frac{1}{P_{atm}}\right)^{\mathrm{CR}_{ir}} \quad . \qquad (32)$$

The equilibrium constant in terms of partial pressures is converted to the molar based constant required for Equation (24) using the following expression,

$$K_{C_r} = K_{pr}/\left(\left(R_p T\right)^{\sum_{i=1}^{N} \mathrm{CR}_{ir}}\right) \qquad (33)$$

Once the forward reaction rate constants are known, the backward reaction rate constant is,

$$K_{br} = K_{fr}/K_{C_r} \qquad (34)$$

2.6 Two Equation Model of Turbulence

In general, there are several types of low-Reynolds number k-ε models and the major distinguishing feature of these models is the handling of the no-slip wall singularity that exists in the destruction term of the specific dissipation equation. The most common practice is to use a modified form of the transport equations, where a fictitious dissipation($\bar{\varepsilon}$) rather than the real dissipation(ε) is used. The fictitious dissipation is usually defined as $\bar{\varepsilon} = \varepsilon - \varepsilon_{wall}$. There are three types of low-Reynolds number k-ε models. The first type uses flowfield characteristics, such as curvature of the turbulent kinetic energy($\varepsilon_{wall} = 2\nu(d^2k/dy^2)^2$), to satisfy the near wall asymptotic behavior, while at same time avoiding the singularity caused by the no-slip boundary condition. The early k-ε model formulation proposed by Launder and Sharma[24] is typical of this type of formulation. The second type of low-Reynolds number k-ε model, proposed by Chien,[25] uses the wall normal distance($\varepsilon_{wall} = 2\nu k/y_n^2$) and Van-Driest type damping to develop the necessary near wall characteristics. The third type of low-Reynolds number formulation makes no attempt to modify the transport equation; rather, finite near-wall characteristics are used as boundary conditions. The low Reynolds number model proposed by Shih[12]is typical of the models that avoid the zero boundary condition and therefore the singularity. All three employ additional wall damping to satisfy the near wall physics.

In practice, the low-Reynolds number k-ε models are easier to implement numerically than high Reynolds number models with wall function boundary conditions for complex geometries because of the simple boundary conditions. However, all of the turbulence models have limitations imposed by the numerical implementation and physics.[28] Therefore, the two-equation model solver in ALLSPD-3D has been designed to be generic and modular.

Presently, the low-Reynolds number model developed by Shih, et al.[12] of CMOTT has been incorporated into ALLSPD-3D as the default turbulence model. However, the modular design of the numerical implementation allows the exact form of the low Reynolds number characteristics to be changed with only minor modifications. Furthermore, several additional numerical options are incorporated to address the numerical stiffness caused by the high near-wall grid resolution. The typical form of the source term required for the Shih et. al. model is given by the following,

$$\begin{bmatrix} S_k \\ S_\varepsilon \end{bmatrix} = \begin{bmatrix} P_k - \rho\varepsilon \\ C_{\varepsilon 1}\frac{\varepsilon}{k}P_k - C_{\varepsilon 2}f_\varepsilon\frac{\rho\varepsilon^2}{k} \end{bmatrix} + \Lambda \tag{35}$$

where the exact turbulent production, P_k, is,

$$P_k = \overline{\rho\tau_{ij}}\left[\frac{\partial u_i}{\partial x_j}\right] \tag{36}$$

$$\overline{\rho\tau_{ij}} = \mu_t\left[\left[\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right] - \frac{2}{3}\left[\frac{\partial u_k}{\partial x_k}\delta_{ij}\right]\right] - \frac{2}{3}\bar{\rho}k\delta_{ij} \tag{37}$$

Here a linear stress-strain relationship is used to model the Reynolds' stress components. The low-Reynolds number corrections required for the source term are given by the following expression,

$$\Lambda = \begin{bmatrix} 0 \\ -2\mu\mu_t(\nabla^2 u)^2 \end{bmatrix} \tag{38}$$

The modeling constants f_ε, $C_{\varepsilon 1}$, and $C_{\varepsilon 2}$ are summarized in Table 2 .

Model	$C_{\varepsilon 1}$	$C_{\varepsilon 2}$	C_μ	σ_κ	σ_ε	f_ε
Shih et. al.	1.44	1.92	0.09	1.00	1.30	$1\text{-}0.22\exp(-R_t^2/36)$

TABLE 2. Turbulence Model Parameters

The eddy viscosity is defined by

$$\mu_t = C_\mu \sqrt{\left(1 - exp\left(a_1 R_k + a_3 R_k^3 + a_5 R_k^5\right)\right)} \rho \frac{k^2}{\varepsilon} \qquad (39)$$

and the modeled damping coefficients are $a_1 = -1.7\text{x}10^{-3}$, $a_3 = -1\text{x}10^{-9}$, and $a_5 = -5\text{x}10^{-10}$.

The eddy viscosity constant can also be varied as a function of the mean shear as suggested by Shih. The formulation of a variable eddy viscosity constant has been developed to satisfy realizability criteria and has been shown to be more accurate for non-homogeneous flows and wall bounded flow situations.[12]

$$C_\mu = min\left(0.09, \frac{2/3}{A_0 + (k/\varepsilon)\sqrt{2S_{ij}S_{ij}}}\right) \text{ where } S_{ij} = \left(\frac{1}{2}\left[\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right] - \frac{1}{3}\left[\frac{\partial u_k}{\partial x_k}\delta_{ij}\right]\right) \qquad (40)$$

2.7 Transport Property Model

All of the transport properties used in ALLSPD-3D have been derived from the database of the physical property information developed by Gordon and McBride.[16] However, in order to reduce complexity, property curves for the diffusion and conductivity coefficients are converted into single forth order polynomial curves. The typical diffusion property curves for the i^{th} species are defined by the following expressions,

$$\mu_i = \sum_{k=0}^{4} C_{\mu_i}(k) T^k \qquad \kappa_i = \sum_{k=0}^{4} C_{\kappa_i}(k) T^k \qquad (41)$$

The $C_{\mu_i}(k)$ constants have the dimensional property of [kg/(m-s)/K^k], and $C_{\kappa_i}(k)$ constants have the dimensional property of [Joules/(m-s-K)/K^k]. Once the individual species viscosities are found, the mixture viscosity and conductivity are computed using Wilke's mixing rule,[32]

$$\mu = \sum_{i=1}^{N} \mu_i \Big/ \left(\frac{1}{X_i} \sum_{j=1}^{N} X_j \phi_{ij} \right) \qquad \kappa = \sum_{i=1}^{N} \kappa_i \Big/ \left(\frac{1}{X_i} \sum_{j=1}^{N} X_j \phi_{ij} \right) \tag{42}$$

Here, the inter-collisional parameter(ϕ) between species i and j is defined as

$$\phi_{ij} = \left(2\sqrt{8\left(1 + \frac{MW_i}{MW_j}\right)} \right)^{-1} \left(1 + 4\sqrt{\frac{MW_j}{MW_i}} \sqrt{\frac{\mu_i}{\mu_j}} \right)^2 \tag{43}$$

The inter-species diffusion, between the species *i* and *j*, is modeled using the Chapman-Enskog[34] formula for dilute gases,

$$D_{ij} = 0.0018583 \sqrt{\frac{1}{WM_i} + \frac{1}{WM_j}} \left(\frac{1}{\sigma^2{}_{ij}} \right) \left(\frac{T^{1.5}}{P\Omega_{ij}} \right) \tag{44}$$

where the effective collision integral factor, Ω_{ij}, and the effective temperature, T_D, are approximated by

$$\begin{gathered} \Omega_{ij} = \frac{1}{T_D^{0.145}} + \frac{1}{\left(T_D + 0.5\right)^2} \\ T_D = \frac{T}{\sqrt{T_\varepsilon(i)\, T_\varepsilon(j)}} \qquad \sigma_{ij} = \frac{1}{2}\left(\sigma_i + \sigma_j\right) \end{gathered} \tag{45}$$

Once, the individual species diffusivity properties have been determined, the overall diffusion coefficients are computed by lumping all of the species together into a mixture and then computing the binary diffusion between the ith species and the mixture. An approximate form of the diffusion coefficient is as follows,

$$D_{im} = \left(\left(1 - X_i\right) \Big/ \left(\sum_{j \neq i}^{N} \frac{X_j}{D_{ij}}\right)\right) \tag{46}$$

Once, the binary diffusion coefficients, $\boldsymbol{D}_{ij}$, for all of the species combinations are known, the diffusion velocity of each species is found using Fick's law. The species diffusion velocity components, $\hat{u}_k\left(Y_i\right)$, are defined as

$$\hat{u}_k\left(Y_i\right) = -D_{im}\frac{\partial Y_i}{\partial x_k} \tag{47}$$

The turbulent diffusion of momentum, mass and heat are modeled based on the assumption that the eddy diffusivity is directly proportional to the turbulent diffusivity by a constant, which is the turbulent Schmidt number (Sc_t) for mass diffusion and the turbulent Prandtl number (Pr_t) for thermal diffusion. Once combined with the mean properties, effective diffusion coefficients can be extracted and defined as follows,

$$\begin{gathered} \mu_{\text{eff}} = \mu + \mu_t \qquad \left(D_{im}\right)_{\text{eff}} = D_{im} + \frac{\mu_t}{\rho Sct} \\ \kappa_{\text{eff}} = \kappa + \frac{\mu_t C_{p_m}}{Prt} \end{gathered} \tag{48}$$

The turbulent Schmidt number and the turbulent Prandtl number are assumed to be equal to 0.90.

2.8 Thermodynamic Model

The mass-averaged constant pressure specific heat constant (C_{p_i}) is assumed to be a fourth order polynomial function of temperature. The curve fitted constants have been adopted from Gordon and McBride[16] Once the individual species specific heat (C_{p_i}) have been found, the mixture specific heat (C_{p_m}) can be found by weighting over all of the species,

$$C_{p_i} = \sum_{k=0}^{4} Cp_i(k)T^k \qquad C_{p_m} = \sum_{i=1}^{N} Cp_iY_i \tag{49}$$

The constants have the dimensional property of [Joules/kg/K^k]. The internal energy of the gas mixture, found by using mixture properties such as pressure, density and enthalpy, is defined as

$$e = \sum_{i=1}^{N} Y_ih_i - \frac{p}{\rho} \qquad h_i = h_{f_i} + \int_{Tref}^{T} Cp_idT \tag{50}$$

where h_{fi} is an integration constant to satisfy the reference temperature condition of Gordon and McBride.[16] Since the specific heats, C_{pi}, are formulated as a function of temperature only, the enthalpy relationship can be easily integrated to yield a polynomial relationship between enthalpy and temperature for the gas mixture. Newton's iteration is used to find the temperature from the predicted internal energy of the gas mixture. The pressure of the gas mixture is determined from Dalton's law of partial pressure,

$$p = \rho R_u T \sum_{i=1}^{N} \frac{Y_i}{MW_i} \tag{51}$$

2.9 Turbulent Chemistry Model

The influence of turbulent intermittence on the finite rate chemistry is modeled using a simple eddy dissipation concept. The simple formulation of Magnussen and Hjertager[13] is employed where the intermittency of the turbulent flame is modeled in terms of rates of dissipation of eddies. Here the rate of eddy dissipation is related to the mean concentrations of fuel species and the turbulent time scales, where the turbulent time scale is defined as the ratio of the turbulent kinetic energy to specific dissipation. The eddy dissipation concept offers one distinct advantage over the eddy break up model concept of Spalding[36] in that the modeled term is directly related to the mean species quantities and does not require additional modeling of species variance terms($\sqrt{Y_s^{\prime 2}}$). This turbulent chemistry model is a first order correction that has been shown to be accurate in combustor applications.[31]

The species source term modeling the mass transfer between the species shown in Equation (24) is replaced with the following,

$$\varpi_i = C_{ebm}\rho Y_i\frac{\varepsilon}{k} \qquad \omega_i = \mathrm{Min}\left(\omega_i^{\mathrm{Laminar}}, \varpi_i^{\mathrm{fuel}}, \varpi_i^{\mathrm{oxy}}\right) \tag{52}$$

III : Numerical Methods

Equation(7) is the final gas-phase governing equation to be solved numerically. As mentioned in the introduction, solving the coupled three-dimensional equations numerically with "many" species using a reasonable number of grid points (say, a quarter million) in a fully coupled manner is impractical. The main problem with using the fully coupled approach, like the one in Chen,et. al.[7] is that it requires a large amount of storage for the left-hand-side (LHS) coefficient matrix. Although great effort has been expended to take advantage of several storage saving tactics, this problem has not been fully solved. As the number of species and the grid density increase, this problem becomes more difficult to resolve.

Although it is believed that the fully coupled, strongly implicit algorithm is the best choice (as far as the stability, robustness and convergence are concerned) for stiff problems typical of combustion flows, it is not very economical or affordable for the practical complex problems found in industry. Other partially implicit and decoupled (sequential) schemes, such as TEACH, can experience difficulties in handling reacting flows, especially with sprays. There is currently a wide spectrum of numerical schemes, whose applicability depends on the specific problem of interest. The advantages of a particular scheme may be the disadvantages of others. In the area of three-dimensional combustion flow simulation, however, the freedom of choice is very narrow.

The approach in ALLSPD-3D employs a fully coupled, strongly implicit algorithm with a slight modification to allow more efficient computation for three-dimensional reacting flows. The partially decoupled procedure described here is designed to accommodate more species and more grid points in three-

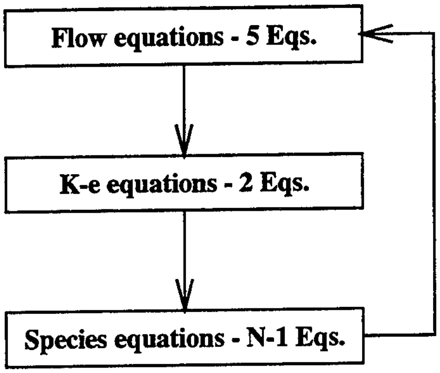


FIGURE 1. **Partially Decoupled Procedure**

dimensional combustion calculations. It is believed that this approach represents a reasonable trade-off between various numerical approaches available for the computation of finite-rate combustion flows. As the growth of computer technology in terms of memory and speed progresses, the advantages of the current approach will become more pronounced.

In ALLSPD-3D, the physical time (second term in Equation(7)) term has been eliminated and the solution marches forward in pseudo-time to steady state. The fully coupled governing equations, Equation(7), are partially decoupled into three sets of equations, i.e., five flow equations, the κ-ε turbulent equations and (*N*-1) species equations. The numerical scheme used in this code solves each set of equations in a coupled manner and iteratively. This process is depicted in Figure 1 .

The governing equations for each set are similar and have the following general form:

$$\Gamma^{(m)}\frac{\partial}{\partial\tau^*}(\hat{Q}^{(m)})+\frac{\partial}{\partial\xi}\left(\tilde{E}^{(m)}-\tilde{E}_v^{(m)}\right)+\frac{\partial}{\partial\eta}\left(\tilde{F}^{(m)}-\tilde{F}_v^{(m)}\right)+\frac{\partial}{\partial\zeta}\left(\tilde{G}^{(m)}-\tilde{G}_v^{(m)}\right) \quad (53)$$

$$=\tilde{H}_c^{(m)}+\tilde{H}_l^{(m)}+\tilde{D}_c^{(m)}$$

where m=1 for flow equations, m=2 for turbulent equations, and m=3 for species equations. For example,

$$\hat{Q}^{(1)} = \left(\frac{1}{J}(p_g, u, v, w, h)\right)^T \tag{54}$$

and

$$\hat{Q}^{(2)} = \left(\frac{1}{J}(k, \varepsilon)\right)^T \qquad \hat{Q}^{(3)} = \left(\frac{1}{J}(Y_1 ... Y_{N-1})\right)^T \tag{55}$$

$$\Gamma^{(1)} = \begin{bmatrix} \frac{1}{\beta} & 0 & 0 & 0 & 0 \\ \frac{u}{\beta} & \rho & 0 & 0 & 0 \\ \frac{v}{\beta} & 0 & \rho & 0 & 0 \\ \frac{w}{\beta} & 0 & 0 & \rho & 0 \\ \frac{H}{\beta} - 1 & \rho u & \rho v & \rho w & \rho \end{bmatrix} \tag{56}$$

$$\Gamma^{(2)} = \begin{bmatrix} \rho & 0 \\ 0 & \rho \end{bmatrix} \qquad \Gamma^{(3)} = \begin{bmatrix} \rho & . & . & . \\ . & \rho & 0 & . \\ . & 0 & . & . \\ . & . & . & \rho \end{bmatrix} \tag{57}$$

Other terms in Equation(53) are straightforward to obtain from Equation(7) and are not listed here. The spray source terms, $\tilde{H}_l^{(m)}$, used to model the mass transfer are described in detail in Section 3.3.

$$H_l^{(1)} = \frac{1}{J}\begin{pmatrix} \sum_p n_p \dot{m}_p \\ \sum_p n_p \dot{m}_p u_p - \frac{4\pi}{3}\rho_p r_p^3 n_p \frac{du_p}{dt} \\ \sum_p n_p \dot{m}_p v_p - \frac{4\pi}{3}\rho_p r_p^3 n_p \frac{dv_p}{dt} \\ \sum_p n_p \dot{m}_p w_p - \frac{4\pi}{3}\rho_p r_p^3 n_p \frac{dw_p}{dt} \\ \sum_p n_p \dot{m}_p h_{fs} - 4\pi r_p^2 n_p h \Delta T \end{pmatrix}$$

$$H_l^{(2)} = \frac{1}{J}\begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad H_l^{(3)} = \frac{1}{J}\begin{pmatrix} \sum_p n_p \dot{m}_p \\ 0 \\ 0 \\ . \\ . \\ 0 \end{pmatrix}$$

The $\tilde{D}^{(m)}$ term that appears in Equation(53) represents a source term arising from the partially decoupled procedure and is expressed as,

$$\tilde{D}_c^{(1)} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \tilde{D}_c^{(2)} = -\frac{1}{\beta}\frac{\partial}{\partial \tau^*}\left(\frac{p_g}{J}\right)\begin{bmatrix} k \\ \varepsilon \end{bmatrix} \qquad \tilde{D}_c^{(3)} = -\frac{1}{\beta}\frac{\partial}{\partial \tau^*}\left(\frac{p_g}{J}\right)\begin{bmatrix} Y_1 \\ Y_2 \\ . \\ . \\ Y_N \end{bmatrix}$$

3.1 LU Scheme

After linearization and applying a first-order time differencing to the pseudo-time term, Equation(53) can be expressed in the following form,

$$\left[\Gamma^{(m)} - \Delta\tau^{*}\left(T^{(m)} + \left(\frac{\partial}{\partial\xi}A^{(m)} - \frac{\partial}{\partial\xi}R_{\xi\xi}^{(m)}\frac{\partial}{\partial\xi}\right) + \left(\frac{\partial}{\partial\eta}B^{(m)} - \frac{\partial}{\partial\eta}R_{\eta\eta}^{(m)}\frac{\partial}{\partial\eta}\right) + \left(\frac{\partial}{\partial\zeta}C^{(m)} - \frac{\partial}{\partial\zeta}R_{\zeta\zeta}^{(m)}\frac{\partial}{\partial\zeta}\right)\right)\right]\Delta\hat{Q}^{(m)} = -\Delta\tau^{*}(R^{(m)})^{p} \tag{58}$$

where,

$$\left(R^{(m)}\right)^{p} = \frac{\partial\left(\tilde{E}^{(m)} - \tilde{E}_{v}^{(m)}\right)}{\partial\xi} + \frac{\partial\left(\tilde{F}^{(m)} - \tilde{F}_{v}^{(m)}\right)}{\partial\eta} + \frac{\partial\left(\tilde{G}^{(m)} - \tilde{G}_{v}^{(m)}\right)}{\partial\zeta} - \tilde{H}_{c}^{(m)} - \tilde{H}_{l}^{(m)} - \tilde{D}_{c}^{(m)} - \tilde{D}_{4}^{(m)}$$

where the superscript p denotes the previous iteration level, and $\boldsymbol{S}^{(m)}$ and $\boldsymbol{T}^{(m)}$ are the Jacobian matrices for chemical and turbulent source terms, respectively. $\boldsymbol{A},^{(m)}$ $\boldsymbol{B}^{(m)}$ and $\boldsymbol{C}^{(m)}$ are the inviscid term Jacobians and $\boldsymbol{R}\xi\xi^{(m)}$, $\boldsymbol{R}\eta\eta^{(m)}$ and $\boldsymbol{R}\zeta\zeta^{(m)}$ are the viscous Jacobians and $\Delta\hat{Q}^{p+1} = \Delta\hat{Q}^{p} + \Delta\hat{Q}^{(m)}$. The expressions for these Jacobians are given in Appendix A. $\boldsymbol{D}^{(m)}{}_{4}$ is the fourth order dissipation term that will be defined later. The lower-upper symmetric Gauss-Seidel (LU-SGS) scheme described here was originally developed by Yoon and Jameson[19] for Euler and Navier-Stokes calculations without preconditioning. With the current time preconditioning, the LU-SGS scheme has to be modified to incorporate the preconditioning matrix. Consider Equation(53), for example. The LHS inviscid terms, which include the preconditioning matrix, need to be modified. For example, in the ξ direction,

$$\frac{\partial A^{(m)}}{\partial\xi} = \frac{\partial}{\partial\xi}\left(\Gamma^{(m)}\Gamma^{(m)^{-1}}A^{(m)}\right) = \frac{\partial}{\partial\xi}\left(\Gamma^{(m)}\hat{A}^{(m)}\right) \tag{59}$$

where $\hat{A}^{(m)} = \left(\Gamma^{(m)^{-1}}A^{(m)}\right)$. $\hat{A}^{(m)}$ is then split into two parts,

$$\begin{aligned}\hat{A}^{(m)} &= \frac{1}{2}\left(\hat{A}^{(m)} + \alpha\,|\lambda_A|^{(m)}_{max} I\right) + \frac{1}{2}\left(\hat{A}^{(m)} - \alpha\,|\lambda_A|^{(m)}_{max} I\right)\\ &= \hat{A}^{(m)^+} + \hat{A}^{(m)^-}\end{aligned}$$

where $|\lambda_A|^{(m)}_{max}$ is the absolute value of the maximum eigenvalue in the ξ direction at each grid point, α is a parameter greater than 1, and I is the unit matrix. The term $\partial A^{(m)}/\partial\xi$ in Equation(59) is then approximated by the following one-sided operator,

$$\frac{\partial A^{(m)}}{\partial\xi} \approx \frac{\partial^-}{\partial\xi}\left(\Gamma^{(m)}\hat{A}^{(m)^+}\right) + \frac{\partial^+}{\partial\xi}\left(\Gamma^{(m)}\hat{A}^{(m)^-}\right) \tag{60}$$

where

$$\begin{aligned}\Gamma^{(m)}\hat{A}^{(m)^\pm} &= \frac{1}{2}\Gamma^{(m)}\left(\hat{A}^{(m)} \pm \alpha\,|\lambda_A|^{(m)}_{max} I\right)\\ &= \frac{1}{2}\left(\hat{A}^{(m)} \pm \alpha\,|\lambda_A|^{(m)}_{max}\Gamma^{(m)}\right)\end{aligned}$$

The operators, $\partial^-/\partial\xi$ and $\partial^+/\partial\xi$ in Equation(60) are the first order backward and forward differencing operators, respectively. Similar procedure can be applied to $\partial B^{(m)}/\partial\eta$ and $\partial C^{(m)}/\partial\zeta$. It can be easily shown that the effect of the above eigenvalue augmentation is equivalent to adding a second order central-differencing dissipation term (scaled by the local maximum eigenvalues) to the implicit operator. Therefore, the discretized governing equations for LU-SGS scheme can be written as follows,

$$\begin{aligned}&\left[\Gamma^{(m)} - \Delta\tau^*\left(T^{(m)} + \left(\frac{\partial}{\partial\xi}A^{(m)} - \frac{\partial}{\partial\xi}R^{(m)}_{\xi\xi}\frac{\partial}{\partial\xi}\right) + \left(\frac{\partial}{\partial\eta}B^{(m)} - \frac{\partial}{\partial\eta}R^{(m)}_{\eta\eta}\frac{\partial}{\partial\eta}\right) + \right.\right.\\ &\left.\left.\left(\frac{\partial}{\partial\zeta}C^{(m)} - \frac{\partial}{\partial\zeta}R^{(m)}_{\zeta\zeta}\frac{\partial}{\partial\zeta}\right)\right) + D_2^{(m)}\right]\Delta\hat{Q}^{(m)} = -\Delta\tau^*\,(R^{(m)})^p\end{aligned}$$

where the second and fourth order dissipation terms are defined as,

$$D_2^{(m)} = -0.5\alpha\left[\frac{\partial^2}{\partial\xi^2}\left(|\lambda_A|\Gamma^{(m)}\right)+\frac{\partial^2}{\partial\eta^2}\left(|\lambda_B|\Gamma^{(m)}\right)+\frac{\partial^2}{\partial\zeta^2}\left(|\lambda_C|\Gamma^{(m)}\right)\right]$$

$$D_4^{(m)} = \sigma\Gamma^{(m)}\left[|\lambda_A|\frac{\partial^4}{\partial\xi^4}(\hat{Q}^{(m)})+|\lambda_B|\frac{\partial^4}{\partial\eta^4}(\hat{Q}^{(m)})+|\lambda_C|\frac{\partial^4}{\partial\zeta^4}(\hat{Q}^{(m)})\right]$$

where $|\lambda_A^{(m)}|$, $|\lambda_B^{(m)}|$ and $|\lambda_C^{(m)}|$ are the absolute values of the maximum eigenvalues in the ξ, η and ζ directions, respectively. The parameter α and σ are used to define the strength of the dissipation. Here, the eigenvalues for the flow equations are identical to those previously defined. The eigenvalues for turbulent and species equations are simply the contravariant velocity components. The LU-SGS scheme utilizes the following factorization procedure to obtain the solution,

$$M\Delta\hat{Q} = b \qquad (61)$$

$$(L+D)D^{-1}(U+D)\Delta\hat{Q} = b \qquad (62)$$

Equation(62) can be solved in the following steps,

Step 1 $\quad (L+D)\Delta\hat{Q}^* = b$

Step 2 $\quad (U+D)\Delta\hat{Q} = D\left(\Delta\hat{Q}^*\right)$

Step 3 $\quad \Delta\hat{Q}^{p+1} = \Delta\hat{Q}^p + \Delta\hat{Q}$

where ***M*** and ***b*** are the LHS and RHS of Equation(53), respectively. The ***L***, ***U*** and ***D*** are the non-zero, off-diagonal lower, upper and diagonal part of matrix ***M***, respectively. The LU-SGS scheme can be vectorized along a constant (i+j+k) oblique planes (see Figure 2). Therefore, the $n\times n$ LHS coefficient blocks are stored only for the two-dimensional oblique plane. Here, n is the number of the equations to be solved.

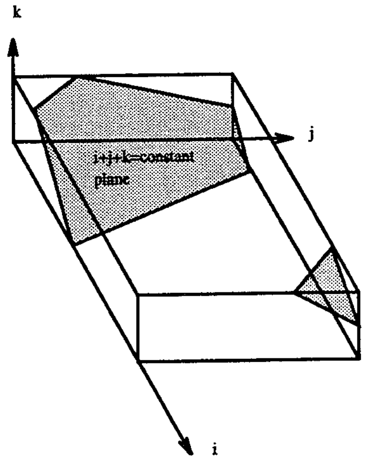


FIGURE 2. Vectorized Oblique Plane

3.2 Mass Conservation

The first set of equations in the partially decoupled procedure is straightforward to solve. There are no numerical differences between this set of equations and those in the fully coupled approach[7] except that only five equations are solved. However, when the turbulent equations or species equations are solved, there is an extra source term, i.e., $\tilde{D}_c^{(2\text{ or }3)}$. These terms consist of a common factor, i.e, $\left(\partial (p_g/J)/\partial\tau^*\right)/\beta$, which has to be evaluated from the continuity equation in order to conserve mass in a numerical sense. Since the continuity equation can be written as,

$$\frac{1}{\beta}\frac{\partial}{\partial\tau^*}\left(\frac{p_g}{J}\right)+\nabla\ \bullet\left(\rho\vec{V}\right)=\sum_p n_p\dot{m}_p$$

The, $\left(\partial (p_g/J)/\partial\tau^*\right)/\beta$ term can be evaluated as:

$$\frac{1}{\beta}\frac{\partial}{\partial\tau^*}\left(\frac{p_g}{J}\right) = -\nabla \bullet\left(\rho\vec{V}\right)+\sum_p n_p\dot{m}_p \qquad (63)$$

3.3 Spray Source Terms

In Equation(53), the liquid-phase interaction with the Navier-Stokes equations is modeled as a source term which accounts for mass, momentum and energy exchanges between the two phases. In order to compute this source term, the liquid-phase governing equations described in Section 2.3 have to be integrated in time along each particle trajectory from a specified initial condition. As a liquid droplet begins its journey from the injection nozzle, its position, velocity, temperature, and size will change according to its interaction with the gas-phase flowfield. In contrast to the Eulerian approach for the gas-phase equations, the Lagrangian treatment for the liquid-phase equations requires interpolation of the flow quantities from the Eulerian grid to the particle positions and re-distribution of the spray source terms from the computed particle positions to the Eulerian grid for the gas-phase calculation. The overall computational time step is carefully chosen since the time step required for the liquid phase calculations is much smaller than that of the flow solver and because the present LU scheme allows the use of a large CFL number. A large number of particle groups are required to statistically represent the spray behavior. Therefore, it is very important to keep the numerical efficiency of the present implicit scheme from being severely degraded by the coupling (or interaction) between the gas and liquid phases. A strategy used in ALLSPD-3D code to overcome this problem,[20]in steady state flows, is described in the following subsections.

3.3.1 Stochastic Process

In the present code, the stochastic separated flow (SSF) model is applied to account for the dispersion effects on spray characteristics. In this model, the gas-phase velocity fluctuations, u_g' $v_g'$and w$_g'$,are generated by randomly sampling a Gaussian probability density distribution having a standard deviation of $\sqrt{(2k)/3}$. The instantaneous velocities are then used to evaluate Equation(11). A fixed number of samplings is conducted for each group of particles. The final spray quantities are obtained by averaging the results (source terms, trajectories,...,etc.) among the total samples. Details of the SSF model can be found in Faeth[14] and Shuen.[21]

3.3.2 Determination of Spray Time Step

For spray combustion calculations, several time scales exist in the flow field which can differ by several orders of magnitude. The chemical reaction time scale is usually very small compared to the rate of evolution of the gas flow. This is also true for spray evaporation. To accurately calculate particle trajectories, size and temperature, the integration time step has to be small. This restriction is especially severe as the droplet becomes smaller and smaller toward the end of its lifetime. For an unsteady problem, the time step for the entire system will be controlled by the smallest time scale. For a steady state calculation, however, the time step for chemical reaction is usually not a problem in the present formulation due to the implicit treatment of the chemical source term. Experiences indicates that for single-phase combustion calculations, the same CFL number usually can be used for both non-reacting and reacting calculations with the present LU method. This ensures that the convergence properties for reacting flow calculations do not deteriorate using the present numerical algorithm. However, the time step for spray equations still remains small and has to be selected (computed) very carefully in order to obtain accurate spray results and to maintain stability. Here we describe the determination of the spray time step. The spray interactions with the gas flow are described in the next section.

The spray time step at any instant along its trajectory is determined from the following time step constraints: (1) droplet velocity relaxation time (t_r), (2) droplet life time (t_l), (3) droplet surface temperature constraint time (t_s), (4) local grid time scale (t_g) and (5) turbulent eddy-droplet interaction time (t_i). The final spray time step (Δt_{spr}) is determined by taking the minimum of t_r, t_l t_s, t_g and t_i to ensure the accuracy and stability of the spray calculations. A factor between 0.1 and 0.5 is used to multiply the selected time step in the current spray calculation. These time steps are described in more detail below.

Droplet Velocity Relaxation time (t_r)

The locally linearized droplet equations of motion, Equation(11), have an exact solution in terms of the local slip velocity with an exponential decay form. The time constant for the exact solution can be expressed as,

$$t_r = \frac{16}{3}\left(\frac{\rho_l}{\rho_g}\right)\left(\frac{r_p^{\,2}}{\nu}\right)(C_D Re_p)^{-1} \tag{64}$$

Droplet Life Time (t_l)

To ensure that the drop size remains positive for the purpose of practical computations, the droplet lifetime at any instant of time is estimated by the following equation,

$$t_l = \frac{\rho_l r_p}{3\dot{m}_p''} \tag{65}$$

Droplet Surface Temperature Constraint Time (t_s)

When the governing equation for the droplet internal temperature distribution, Equation(19), is solved, the temperature solution can become completely incorrect due to the use of an inappropriately large time step. This is particularly important for fuel with a low boiling temperature (close to room temperature). In order to ensure the success of the temperature calculation using Equation(19), the exact solution of the infinite conductivity model (also called the uniform temperature model) is used to estimate the time step for the present vortex model. The temperature equation for the droplet based on this model is

$$\frac{dT_p}{dt} = \frac{6}{\rho_l C_v d_p}\left[\bar{h}(\tilde{T} - T_p) - \dot{m}_p'' h_{fg}\right] \qquad (66)$$

Equation(66) has a exact solution (after local linearization) of the form,

$$\Delta T_p = A\left(1 - e^{-A't}\right) \qquad (67)$$

where $\Delta\ T_p$ is the droplet temperature change within the integration time step, and A, A' and B' are defined as,

$$\begin{aligned} A &= \left(T - T_p - \frac{B'}{A'}\right) \\ A' &= \frac{6}{\rho_l C_v d_p}\bar{h} \\ B' &= \frac{6}{\rho_l C_v d_p}\dot{m}_p'' h_{fg} \end{aligned} \qquad (68)$$

Therefore, a time scale can be obtained based on the above solution if a desired $\Delta\ T_p$ is specified. This time scale is expressed as,

$$t_s = \frac{\ln\left(\dfrac{1}{1 - \dfrac{\Delta T_p}{A}}\right)}{A'} \qquad (69)$$

In ALLSPD-3D spray calculations, $\Delta\ T_p$=3 K is specified for particles just leaving the injector, where they experience a sudden temperature jump, and $\Delta T_p = 3.0K$ for the rest of the spray calculation (toward the end of their lifetime).

Local Grid Time Scale (t_g)

A particle can travel across several grid cells and may experience sudden changes in local gas properties if the time step is too large. This not only causes inaccuracy in the integration, but also increases the difficulty of locating the particle positions. Therefore, a time scale, t_g, is computed to ensure that the distance a particle moves is less than one local cell size in one time step.

Turbulent Eddy-Droplet Interaction Time (t_i)

According to Shuen et al,[22] a particle is assumed to interact with an eddy for a time which is the minimum of either the eddy lifetime or the transit time required for the particle to cross the eddy. These times are estimated by assuming that the characteristic size of an eddy is the dissipation length scale given by,

$$Le = \frac{C_\mu^{3/4} k^{3/4}}{\varepsilon} \tag{70}$$

and the eddy lifetime is estimated as,

$$t_e = \frac{Le}{\sqrt{\frac{2k}{3}}} \tag{71}$$

The transit time of a particle was found using the linearized equation of motion for a particle in a uniform flow,

$$t_t = -\tau \ln\left[1 - \frac{L_e}{\left(\tau\left|\vec{u}'' - \vec{u}_p''\right|\right)}\right] \qquad \text{where} \qquad \tau = \frac{8\rho_l r_p}{3\rho_g C_D \left|\vec{u}'' - \vec{u}_p''\right|} \tag{72}$$

and $\left|\vec{u}'' - \vec{u}_p''\right|$ is the relative velocity at the start of the interaction. When $L_e > \tau\left|\vec{u}'' - \vec{u}_p''\right|$, the linearized stopping distance of the particle is smaller than the characteristic length scale of the eddy, and then Equation(72) has no solution. In this case, the eddy has captured the particle and the interaction time is the eddy lifetime. Therefore,

$$t_i = t_e \qquad \text{if} \qquad L_e > \tau\left|\vec{u}'' - \vec{u}_p''\right|$$

$$t_i = min\,(t_e, t_t) \qquad \text{if} \qquad L_e \le \tau\left|\vec{u}'' - \vec{u}_p''\right|$$

3.3.3 Interaction Between Two Phases

For the gas-phase equations, Equation(53), the presence of the spray appears in the form of a source term, $\boldsymbol{H}_l^{(m)}$. This source term represents the interchange of the mass, momentum and energy between the two phases. As particles are injected into the flow domain, their subsequent behavior (positions, velocities, size and temperature) is affected continuously by the neighboring gas properties and *vice versa*. Usually a large number of spray particles is desirable to accurately predict the spray behavior. However, this requires tremendous computational effort for the spray calculation alone. To minimize the computational time for the spray, the spray source term, $\boldsymbol{H}_l^{(m)}$, need not be updated at every gas-phase iteration (pseudo-time marching). Usually the spray source term is updated every 10 to 20 iterations in ALLSPD-3D spray calculations. When the spray source term is updated, each group of particles is integrated either to the end of its lifetime or until it leaves the computational domain. It should be noted that, for the present steady state spray combustion calculation, the spray time step determined previously is independent of the pseudo-time used for the gas-phase equations, which is determined from the maximum CFL number obtainable according to the local eigenvalues. Therefore, this strategy maintains both the efficiency of the flow solver and the accuracy of the spray Lagrangian integration. This treatment of the gas-liquid interaction is different from those of Raju and Sirignano[23] where time-accurate solutions are of their primary concern. In the present computation, it is assumed that when the droplets impinge on the walls they evaporate completely.

3.4 Pseudo-Time Step

In Equation(58), $\Delta\tau^*$ is the pseudo-time which is determined from the eigenvalue analysis. For the flow set of equations, the eigenvalues are (defined previously),

$$\lambda_A = \frac{1}{2}\left[U_A\left(1+\frac{\hat{\beta}}{c^2}\right) \pm \sqrt{U_A^2\left(1-\frac{\hat{\beta}}{c^2}\right)^2 + 4\hat{\beta}\left(\xi_x^{\ 2}+\xi_y^{\ 2}+\xi_z^{\ 2}\right)}\right]$$

$$\lambda_B = \frac{1}{2}\left[U_B\left(1+\frac{\hat{\beta}}{c^2}\right) \pm \sqrt{U_B^2\left(1-\frac{\hat{\beta}}{c^2}\right)^2 + 4\hat{\beta}\left(\eta_x^{\ 2}+\eta_y^{\ 2}+\eta_z^{\ 2}\right)}\right] \qquad (73)$$

$$\lambda_C = \frac{1}{2}\left[U_C\left(1+\frac{\hat{\beta}}{c^2}\right) \pm \sqrt{U_C^2\left(1-\frac{\hat{\beta}}{c^2}\right)^2 + 4\hat{\beta}\left(\zeta_x^{\ 2}+\zeta_y^{\ 2}+\zeta_z^{\ 2}\right)}\right]$$

where,

$$U_A = \xi_x u + \xi_y v + \xi_z w,$$

$$U_B = \eta_x u + \eta_y v + \eta_z w,$$

$$U_C = \zeta_x u + \zeta_y v + \zeta_z w,$$

and,

$$\hat{\beta} = max\left(\beta, U_{ref}^2\right)$$

The pseudo-time, $\Delta\,\tau^*$ is then calculated by the following equation,

$$\Delta\,\tau^* = CFL/(\lambda^2_A+\lambda^2_B+\lambda^2_C)^{1/2} \qquad (74)$$

where *CFL* is a user-input CFL number and Uref is a reference velocity.

3.5 Boundary Conditions

Boundary conditions in ALLSPD-3D are applied implicitly. Certain boundary conditions make use of ghost cells which are invisible to the user. In particular, grid interface boundaries acquire information for their ghost cells from the neighboring blocks. The grid connection between blocks is assumed to be contiguous and the neighboring volumes should be approximately the same size.

3.5.1 Wall Boundary Condition

3.5.1.1 Mean Equations

The boundary condition applied to the mean equations assumes that the pressure gradient is zero at the walls. Since the grid may be non-orthogonal, the projected normal pressure gradient is found from the grid metrices. For example, on an i-face

$$\frac{\partial p}{\partial n} = \frac{\partial \xi \partial p}{\partial n \partial \xi} + \frac{\partial \eta \partial p}{\partial n \partial \eta} + \frac{\partial \zeta \partial p}{\partial n \partial \zeta}$$

A no-slip wall condition with either an adiabatic or an isothermal condition may be chosen. The adiabatic condition is imposed by setting the enthalpy gradient to zero; similar to the pressure condition described above. To specify a wall temperature, the static enthalpy is determined for the desired wall temperature. The species mass fractions used in this calculation are the lagged values resulting from the previous species iteration.

3.5.1.2.1 Dilution Holes

A special boundary condition has been defined for problems containing injection ports. These areas are assumed to be relatively small, and to have a known mass flow rate and static temperature. The pressure is found from a first order extrapolation from the interior domain. The first step in applying this boundary condition is to determine the actual injection area. This is done by taking the average of the magnitude of the area normal to the four corners for every grid cell within the prescribed bounds.

Next, the change in velocity is found by calculating the change in mass flow rate required to meet the imposed conditions assuming that the density is unchanged from the previous iteration,

$$\Delta V = \frac{\dot{m}_{inj}}{\rho A_{inj}} - (u)^p .$$

The velocity direction comes from the direction cosines prescribed by the user. The boundary condition for the energy equation is set by holding the injected fluid to a constant stagnation enthalpy. This is accomplished by calculating the static enthalpy for the prescribed injected temperature using the thermal properties from the previous iteration,

$$\Delta h = h(T_{inj}) + \frac{1}{2}\left(\frac{\dot{m}_{inj}}{\rho A_{inj}}\right)^2 - h_t^p$$

This differs from the upstream boundary condition in that the mass flow rate is prescribed rather than the velocity.

3.5.1.3.2 Transpiration Cooling (or Bleed)

Walls containing transpiration cooling or bleed have been handled in a manner similar to the injection boundary condition. However, for this boundary condition, only the mass flow is prescribed as described above. The conditions on pressure and temperature are the same as for a wall. In other words, the energy equation can be bounded by either an adiabatic wall or by an isothermal wall option.

3.5.1.4 Turbulence Model Equations

At the wall, the turbulent kinetic energy and turbulent dissipation are bounded by non-zero values determined by Shih, et.al. to maintain realizability,

$$k_{bc} = 0.250\mathrm{U}_\tau^2 \qquad \varepsilon_{bc} = 0.251\frac{\mathrm{U}_\tau^4}{\nu} \tag{75}$$

The above constraint on ε is accurate for very near wall flows but may be difficult to satisfy without grid clustering near the walls. For this reason, an optional boundary condition has been included to relax the grid resolution requirement near walls. For this condition, the gradient of turbulent dissipation given by Equation(75) is replaced with $\partial\varepsilon/\partial\eta \;\; = 0$.

3.5.1.5 Dilution Holes

The values for the injected turbulence field are found from the user specified turbulent intensity

$$\Delta k = \left[\frac{3}{2}\frac{\dot{m}T_{u_{inj}}}{\rho A_{inj}} - k^{p}\right]$$

where T_u is the injected turbulent intensity, and A_{inj} is the total injection area. The assumed injected dissipation is then calculated from the turbulent kinetic energy and a specified length scale. For a dilution hole, the assumed length scale is the square root of the injection area.

$$\Delta\varepsilon = \left[\frac{\left(\frac{3}{2}\frac{\dot{m}T_{u_{inj}}}{\rho A_{inj}}\right)^{\frac{3}{2}}}{\sqrt{A_{inj}}} - \varepsilon^{p}\right]$$

3.5.1.6 Transpiration Cooling (or Bleed)

For this type of boundary condition, the turbulence quantities are taken from the closest interior grid point

3.5.2 Species Continuity Equations

At a wall the gradient of the species mass fractions is set to zero.

3.5.3 Dilution Holes or Transpiration Cooling (or Bleed)

For these boundary conditions the composition of the injected fluid is prescribed by the user and remains unchanged.

3.5.4 Upstream

At an upstream boundary, the stagnation enthalpy is held constant. This is accomplished by retaining the initial values for the three velocity components, species mass fractions, and by holding the inlet temperature constant. Pressure is extrapolated from the interior points. The default extrapolation method is a second order difference. The turbulence quantities are held at their initial definitions as are the species mass fractions.

The default upstream conditions for the turbulent flow scalar quantities are generated using following expressions. The velocity scales and turbulent kinetic energy are computed based on local or reference velocity levels and turbulence intensity levels.

$$k = \frac{3}{2}\left(T_{\mathrm{u}} U_{\mathrm{ref}}\right)^2 \qquad (76)$$

The specific dissipation is modeled using the following three formulations for attached flow situations. The first expression uses that the most fundamental form of the specific dissipation where it is expressed as a function of velocity and length scales. Here, the length scale is assumed to be the Karman's mixing length scale formulation.

$$\varepsilon = 2.4\frac{k^{3/2}}{L_{\mathrm{ref}}} \qquad L_{\mathrm{ref}} = 0.4y_n \qquad (77)$$

The second formulation uses the experimental correlation of Mellor.[33] Here, the experimental data is expressed as a function of eddy viscosity profiles. Therefore, the following expression is used to find the specific dissipation,

$$\varepsilon = \frac{k^2}{\left(\frac{\mu_t}{\mu_l}\right)\frac{\mu_l}{\rho}} \qquad \frac{\mu_t}{\mu_l} = \frac{\left(0.4y^+\right)^4}{\left(\left(0.4y^+\right)^4 + 328\right)} \tag{78}$$

The third method is similar to Equation(77). Here, the length scale definition has been modified to yield the following,

$$\varepsilon = \frac{k^2}{500\nu_l L_{ref}} \tag{79}$$

3.5.5 Downstream

The flow through the downstream boundary is assumed to remain subsonic. An exit pressure is imposed at the grid cell past the exit plane. For multiple block problems, different exit pressures may be set for each block, however care must be taken if the exit pressures vary greatly from the initial pressure(which is set to user inputted value of Pref over the exit face). If an exit pressure does require a significant departure from the initial value, a large pressure gradient at the exit can be avoided by slowly decreasing the exit pressure over a number of iteration.

The remaining flow quantities are extrapolated from the interior flow domain. A zero order extrapolation method is the default. An *expert user* flag is included in subroutine qplus1.f to change the order of this extrapolation if needed.

3.5.6 Periodic

For the periodic boundary condition, flow quantities are copied from the opposite face into the ghost cells, e.g., the information for the j=1 plane would come from the j=jmax plane. This boundary condition requires only two grid points in the periodic direction and is therefore the best choice for a pseudo-two-dimensional problem.

3.5.7 Symmetry

The symmetry boundary condition copies the flow field from the neighboring interior point to the "ghost cell", e.g., the information for the j=0 plane would come from the j=2 plane. Since ALLSPD-3D requires a minimum of five grid points (including the ghost cells), this option requires the user to supply a minimum of three grid points.

3.5.8 Far Field

The same condition is used as for the downstream boundary condition.

3.5.9 Singularity

For axisymmetric grids, where a plane collapses to a single line, a special boundary condition has been implemented. This boundary must be applied for the k=1 boundary. The solution domain will not include the points along the singularity, so that the centerline values must be found by extrapolation from the interior points. A check is made for internal blockages at the centerline so that the appropriate wall boundary conditions (see above) can be applied to the turbulence equations.

IV : Input/Output Files

4.1 Input/Output

The input and output file structure is explored in this section. Four user or GUI developed input files are required to execute the ALLSPD-3D code. To run the ALLSPD-3D code, users need the grid file, *grid.dat*, block link file, *gridlink.dat*, property data file, *prop.dat*, and the flow data file, *allspd.dat*. A typical *grid.dat* read format is shown in Figure 3. For cases where users would like to provide the initial conditions for the flow variables within the whole domain, the user-defined initial condition file, *uvwt.dat*, has to be generated. The structure of these input files will be discussed along with several examples. The *gridlink.dat* file contains the grid size, connectivity and boundary condition information. A typical gridlink file is shown in Figure 4. A detailed example is presented in Table 3 . A typical *allspd.dat* file is shown in Figure 5 . Various options are summarized in Table 3 . The ALLSPD-3D property input file, *prop.dat*, is arranged in three major groups: Thermal Property Information, Finite Rate Chemistry Information and Spray Property Information. They are arranged in top-down groups as shown in Figure 6. The requirements of each group's input information is summarized later. For liquid spray problems, users can provide their own spray information data file, *spray.dat*, instead of using the default spray particle distribution generator.

For the output files, the most important files are the *PLOT3D* format solution files which can be examined using the NASA *PLOT3D* plotting package[40] or other post-processors which accept this file format. If a liquid spray is initiated, the particle trajectories, temperatures, sizes are also written to a single file for post-processing. A restart file for subsequent runs may also be generated and is usually needed for running large combustion problems.

4.2 Input Data Files

4.2.1 Grid Data: *grid.dat*

The grid is generated separately by the user and read into the code by the subroutine MESH. The default grid data file name is *grid.dat*. The format for the grid file is consistent with *PLOT3D*'s multi-block format. It is generated as a free formatted file and therefore is portable. There are two formats available for *grid.dat*. One is a non-blanked file, and the other is an Iblanked. The Iblanked file requires an extra set of Iblanked data (0 or 1 for each grid point) in addition to the usual *x*, *y*, and *z* data. For most users, a non-Iblanked format is preferred. The input flag for either Iblanked or non-Iblanked *grid.dat* file is in *gridlink.dat* that will be described later. The format for *grid.dat* is listed as follows.

For non-Iblanked format:

```
read(iunit,*)nblk
read(iunit,*)(imax(n),jmax(n),kmax(n),n=1,nblk)
do n=1,nblk
read(iunit,*)
(((x(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((y(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((z(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n))
end do
```

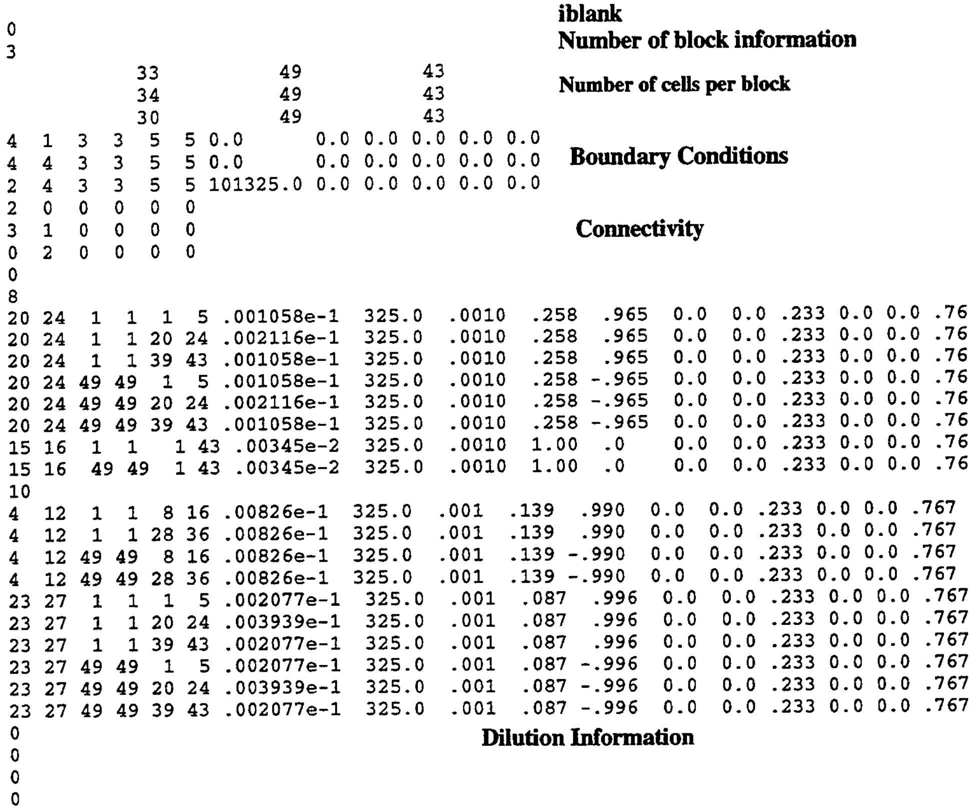
For Iblanked format:

```
read(iunit,*)nblk
read(iunit,*)(imax(n),jmax(n),kmax(n),n=1,nblk)
do n=1,nblk
read(iunit,*)
(((x(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((y(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((z(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((iblk(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n))
end do
```

FIGURE 3. **Read file format for *grid.dat***

4.2.2 Multi-Block Data: *gridlink.dat*

The multi-block connection information is defined in *gridlink.dat* and is read by the subroutine MESH. The input variables are described in Table 3.



```
                                                       iblank
0
3                                                      Number of block information
           33          49          43
           34          49          43                  Number of cells per block
           30          49          43
4  1  3  3  5  5 0.0       0.0 0.0 0.0 0.0 0.0
4  4  3  3  5  5 0.0       0.0 0.0 0.0 0.0 0.0  Boundary Conditions
2  4  3  3  5  5 101325.0 0.0 0.0 0.0 0.0 0.0
2  0  0  0  0  0
3  1  0  0  0  0                                 Connectivity
0  2  0  0  0  0
0
8
20 24  1  1  1  5 .001058e-1  325.0  .0010  .258  .965  0.0  0.0 .233 0.0 0.0 .76
20 24  1  1 20 24 .002116e-1  325.0  .0010  .258  .965  0.0  0.0 .233 0.0 0.0 .76
20 24  1  1 39 43 .001058e-1  325.0  .0010  .258  .965  0.0  0.0 .233 0.0 0.0 .76
20 24 49 49  1  5 .001058e-1  325.0  .0010  .258 -.965  0.0  0.0 .233 0.0 0.0 .76
20 24 49 49 20 24 .002116e-1  325.0  .0010  .258 -.965  0.0  0.0 .233 0.0 0.0 .76
20 24 49 49 39 43 .001058e-1  325.0  .0010  .258 -.965  0.0  0.0 .233 0.0 0.0 .76
15 16  1  1   1 43 .00345e-2  325.0  .0010  1.00  .0    0.0  0.0 .233 0.0 0.0 .76
15 16  49 49  1 43 .00345e-2  325.0  .0010  1.00  .0    0.0  0.0 .233 0.0 0.0 .76
10
4  12  1  1  8 16 .00826e-1  325.0  .001  .139  .990  0.0  0.0 .233 0.0 0.0 .767
4  12  1  1 28 36 .00826e-1  325.0  .001  .139  .990  0.0  0.0 .233 0.0 0.0 .767
4  12 49 49  8 16 .00826e-1  325.0  .001  .139 -.990  0.0  0.0 .233 0.0 0.0 .767
4  12 49 49 28 36 .00826e-1  325.0  .001  .139 -.990  0.0  0.0 .233 0.0 0.0 .767
23 27  1  1  1  5 .002077e-1  325.0  .001  .087  .996  0.0  0.0 .233 0.0 0.0 .767
23 27  1  1 20 24 .003939e-1  325.0  .001  .087  .996  0.0  0.0 .233 0.0 0.0 .767
23 27  1  1 39 43 .002077e-1  325.0  .001  .087  .996  0.0  0.0 .233 0.0 0.0 .767
23 27 49 49  1  5 .002077e-1  325.0  .001  .087 -.996  0.0  0.0 .233 0.0 0.0 .767
23 27 49 49 20 24 .003939e-1  325.0  .001  .087 -.996  0.0  0.0 .233 0.0 0.0 .767
23 27 49 49 39 43 .002077e-1  325.0  .001  .087 -.996  0.0  0.0 .233 0.0 0.0 .767
0                                         Dilution Information
0
0
0
```

FIGURE 4. Typical *gridlink.dat* file

TABLE 3. Detailed Information about *gridlink.dat* File

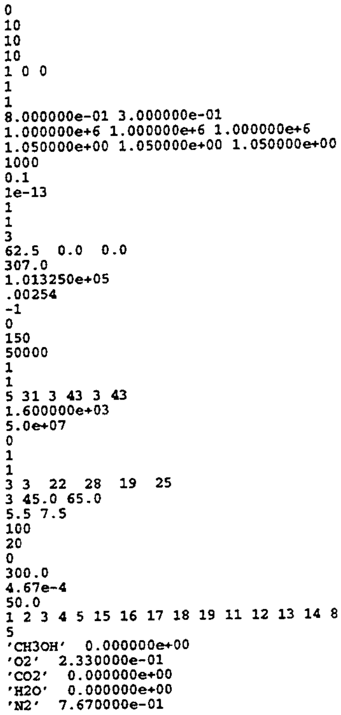
Input Variable	Comments
Iblank	The flag for non-Iblanked or Iblanked *grid.dat* 0 = non-Iblanked *grid.dat* 1 = Iblanked *grid.dat*
(imaxbl(nb),jmaxbl(nb), kmaxbl(nb),nb=1,nblk)	Number of grid points in i, j, and k direction, respectively, in the nb*th* block.
nblk	Total number of blocks
least(nb),lwest(nb), lnorth(nb),lsouth(nb), ltop(nb),lbot(nb),bcin(nb,1) bcin(nb,2),bcin(nb,3), bcin(nb,4),bcin(nb,5), bcin(nb,6), nb=1,nblk	Block boundary control indices, specifying the boundary type in the east, west, north, south, top, and bottom faces of the nb*th* block. The boundary types, **least(nb),lwest(nb),lnorth(nb),lsouth(nb), ltop(nb),lbot(nb),** = 1 upstream (inflow) (all six faces), = 2 downstream (outflow) (all six faces) = 3 solid wall (all six faces), = 4 block interface (all six faces) = 5 symmetry (only for north, south, top, and bottom faces) = 6 periodic boundary (only for north/south, and top/bottom faces) = 7 singularity (only for bottom face, i.e., k=1) = 8 far-field (for external flows) (all six faces) **bcin(nb,1),bcin(nb,2),bcin(nb,3),bcin(nb,4), bcin(nb,5),bcin(nb,6)** are used to assign either pressure or temperature at each faces depending on the boundary types. Zero values are usually used except for those faces that are either wall or exit conditions. For wall boundary faces, zero (0.0) is used to assign an adiabatic wall and desired temperature (°K) is used to assign an isothermal wall. For exit faces, these are used to specify the ambient pressure (N/m^2) that is to remain constant. The corresponding faces are: **bcin(nb,1)** east face, **bcin(nb,2)** west face, **bcin(nb,3)** north face, **bcin(nb,4)** south face, **bcin(nb,5)** top face, **bcin(nb,6)** bottom face
lee(nb),lww(nb),lnn(nb), lss(nb),ltt(nb),lbb(nb) , nb=1,nblk	Block boundary neighboring control indices, indicating the identity of the neighboring block, such as: **lee(nb)=m** - nb*th* block's east side borders m*th* block; zero if nb*th* block's east side is not a block interface. **lnn(nb)=n** - nb*th* block's north side borders n*th* block; zero if nb*th* block's north side is not a block interface, etc ...
intblk	Internal blockage flag = 0 no internal blockage defined; skip the following input data and move directly to the next one for dilution holes (ndil). =1 with internal blockage(s); provide the following input data for internal blockage(s) in each block.
Input Variable	**Comments**

TABLE 3. Detailed Information about *gridlink.dat* File

Input Variable	Comments
nintblk(nb),iintb(nn,nb), iinte(nn,nb),jintb(nn,nb), jinte(nn,nb),kintb(nn,nb), kinte(nn,nb),nn=1, nintblk(nb)	**nintblk(nb)**: number of internal blockages in the nb*th* block.(< 50) (limit defined in parameter file, allspd.h). If zero, skip to the next block; if equal or greater than one (up to **nintblk(nb)**) enter following **iintb(nn,nb), iinte(nn,nb)**: the beginning and ending *i*-index for the nn*th* internal blockage in the nb*th* block. **jintb(nn,nb), jinte(nn,nb)**: the beginning and ending *j*-index for the nn*th* internal blockage in the nb*th* block. **kintb(nn,nb), kinte(nn,nb)**: the beginning and ending *k*-index for the nn*th* internal blockage in the nb*th* block.
ndil(nb),isdil(nb,nn), iedil(nb,nn),jsdil(nb,nn), jedil(nb,nn),ksdil(nb,nn), kedil(nb,nn), dilmass(nb,nn), diltemp(nb,nn), diltin(nb,nn), dilcosx(nb,nn), dilcosy(nb,nn), dilcosz(nb,nn), (dilys(nb,nn,nx), nx=1,ns),nn=1,ndil(nb)	Dilution holes specification. **ndil(nb)** number of dilution holes in the nbth block. If zero, skip to the next block.. **isdil(nb,nn), iedil(nb,nn)**: the beginning and ending *i*-index for the nn*th* dilution hole in the nb*th* block. **jsdil(nb,nn), jedil(nb,nn)**: the beginning and ending *j*-index for the nn*th* dilution hole in the nb*th* block. **ksdil(nb,nn), kedil(nb,nn)**: the beginning and ending *k*-index for the nn*th* dilution hole in the nb*th* block. **dilmass(nb,nn)**: the mass flow rate (*kg/s*) for the nn*th* dilution hole in the nb*th* block. **diltemp(nb,nn)**: the temperature (*K*) for the nn*th* dilution hole in the nb*th* block. **diltin(nb,nn)**: the turbulence intensity for the nn*th* dilution hole in the nb*th* block. **dilcosx(nb,nn), dilcosy(nb,nn), dilcosz(nb,nn)** : the *x*, *y*, *z*-directional cosine for the angle of the cooling velocity for the nn*th* dilution hole in the nb*th* block. **dilys(nb,nn,nx),nx=1,ns**: the mass fractions of each species for the nn*th* dilution hole in the nb*th* block.
ntrn(nb),istrn(nb,nn), ietrn(nb,nn),jstrn(nb,nn), jetrn(nb,nn),kstrn(nb,nn), ketrn(nb,nn), trnmass(nb,nn), trncosx(nb,nn), trncosy(nb,nn), trncosz(nb,nn), (trnys(nb,nn,nx),nx=1,ns), nn=1,ntrn(nb)	Transpiration cooling holes specification. **ntrn(nb)**: number of transpiration cooling holes in the nb*th* block. If zero, skip to the next block.. **istrn(nb,nn), etrn(nb,nn)**: the beginning and ending *i*-index for the nn*th* transpiration cooling hole in the nb*th* block. **jstrn(nb,nn), jetrn(nb,nn)**: the beginning and ending *j*-index for the nn*th* transpiration cooling hole in the nb*th* block. **kstrn(nb,nn), ketrn(nb,nn)**: the beginning and ending *k*-index for the nn*th* transpiration cooling hole in the nb*th* block. **trnmass(nb,nn)**: the mass flow rate (*kg/s*) for the nn*th* transpiration cooling hole in the nb*th* block. **trncosx(nb,nn), trncosy(nb,nn), trncosz(nb,nn)**: the *x*, *y*, *z*-directional cosine for the angle of the cooling velocity for the nn*th* transpiration cooling hole in the nb*th* block. **trnys(nb,nn,nx),nx=1,ns**: the mass fractions of each species for the nn*th* transpiration cooling hole in the nb*th* block.
Input Variable	**Comments**

4.2.3 Flow Data: *allspd.dat*

The flow data and major control of the program execution is done in the *allspd.dat* and is read by the subroutine INPUT. Each input is described in Table 4.



```
0
10
10
10
1 0 0
1
1
8.000000e-01 3.000000e-01
1.000000e+6 1.000000e+6 1.000000e+6
1.050000e+00 1.050000e+00 1.050000e+00
1000
0.1
1e-13
1
1
3
62.5  0.0  0.0
307.0
1.013250e+05
.00254
-1
0
150
50000
1
1
5 31 3 43 3 43
1.600000e+03
5.0e+07
0
1
1
3 3  22  28  19  25
3 45.0 65.0
5.5 7.5
100
20
0
300.0
4.67e-4
50.0
1 2 3 4 5 15 16 17 18 19 11 12 13 14 8
5
'CH3OH'  0.000000e+00
'O2'  2.330000e-01
'CO2'  0.000000e+00
'H2O'  0.000000e+00
'N2'  7.670000e-01
```

FIGURE 5. **Typical Listing of the *allspd.dat* Input File**

TABLE 4. Detailed Layout of ***allspd.dat*** File

Input Data	Input Variable	Comments
0	**iread**	= 0 a fresh run = 1 a restart run
10	**ntmax**	Total number of iterations (psuedo-time steps)
10	**nplot**	Every *nplot* number of iterations the plot3d files are generated, and these files are overwritten every next *nplot* number of iterations
10	**nsave**	Every *nsave* number of iterations the restart file is generated, and this file is overwritten every next *nsave* number of iterations
1, 0,0	**iturb, icmu, idisbc**	turbulence model switch, = 0 laminar flow = 1 turbulent flow, eddy viscosity coefficient = 0 constant $C\mu$ in turbulence model (=0.09) = 1 variable $C\mu$ in turbulence model idisbc = 0 Dirichlet wall boundary condition for ε equation. = 1 Neumann wall boundary condition for ε equation.
1	**icomb**	chemistry switch = 0 non-reacting flow = 1 reacting flow
1	**iartv**	Explicit artificial viscosity switch = 0 no artificial damping = 1 with artificial damping
8.000e-01 3.000e-01	**vis4,** **vis2**	vis4 = 4th order damping coefficient vis2 = 2nd order damping coefficient
1e6, 1e6,1e6	**CFL**	cfl(1) = CFL number for the flow equations cfl(2) = CFL number for the κ-ε equations cfl(3) = CFL number for the species equations
1.05, 1.05, 1.05	**precfac**	precfac(1) = pre-conditioning factor for LU scheme for the flow equations precfac(2) = pre-conditioning factor for LU scheme for the κ-ε equations precfac(3) = pre-conditioning factor for LU scheme for the species equation

TABLE 4. Detailed Layout of *allspd.dat* File

Input Data	Input Variable	Comments
1000	**nswitch**	For some difficult flows, a smaller CFL number initially is helpful to start the calculation. The CFL number can then be ramped up as the flow develops. The CFL number is calculated using the following formula: CFL=CFLxfactau where the factor factau has initial value of facini (defined next) and is linearly increased to 1 over **nswitch** iterations.
0.1	**facini**	The initial value used in conjunction with the above nswitch. **facini** is usually set to 0.1.
1e-7	**bond**	The convergence criterion, say, 1.0e-7 for example
1	**nkecall**	The turbulence equations are updated (called) every **nkecall** flow iterations. Say, if nkecall=10, the turbulence equations are updated every 10 flow iterations
1	**nscall0**	The species equations are updated (called) every **nscall0** flow iterations. Say, if nkecall=10, the turbulence equations are updated every 10 flow iterations
1	**iuvwt**	Initial condition controlling flag = -1 only inlet faces have non-zero flow specified by **uin,vin,** and **win** described next. = 0 uniform flow initialization for the entire domain except for the walls. The flow is defined by **uin,vin,** and **win** described next. = 1 user provides the initial flow condition for the entire domain. In this case the flow data defined here (**uin,vin, win**) will be used only as reference. The description for the data file, *uvwt.dat*, is in another section. = 2 this is used only for easy specification of the swirling inlet condition. The file name is the same as in iuvwt=1 case that is *uvwt.dat*. The format of this file for this case will be described later.
62.5 0.0 0.0	**uin, vin, win**	Initial velocities. These are used to assign initial flow velocity for the entire domain or inlet only, depending on the value of **iuvwt** above. They are used to calculate the inlet flow Reynolds number also. uin = x component of velocity (m/s) vin = y component of velocity (m/s) win = z component of velocity (m/s)
307.0	**Tref**	Reference temperature (K) for the flow domain. This is used to assign initial flow temperature for the entire domain. It is used to calculate the inlet flow Reynolds number also. However, the initial temperature will be overwritten by the temperature defined in *uvwt.dat* if iuvwt=1, or 2.
1.013250e+05	**Pref**	Reference pressure (N/m^2) for the flow domain. This is used to assign initial flow pressure for the entire domain except along outlet faces. It is used to calculate the inlet flow Reynolds number also. However, the initial pressure will be overwritten by the pressure defined in *uvwt.dat* if iuvwt=1, or 2.

TABLE 4. Detailed Layout of ***allspd.dat*** File

Input Data	Input Variable	Comments
.00254	**lref**	A reference length scale(m) used to calculate the inlet Reynolds number and turbulence quantities.
-1	**iebm**	Eddy break-up model for turbulence-combustion closure. = 0 no eddy break-up model = -1 with eddy break-up model
0	**iequate**	A flag to compute the reverse/forward reaction constant in reacting flow calculation. = 0 skip the reverse/forward reaction constant calculation = 1 perform the reverse reaction constant calculation = -1 perform the forward reaction constant calculation
150	**nt1igt**	Ignition energy is added to the gas phase energy equation starting at the time step of **nt1igt**.
50000	**nt2igt**	Ignition energy is turned off from the gas phase energy equation starting after the time step of **nt2igt**.
1	**noigt**	Number of ignition sources, (if = 0, skips to next input of **toff**)
1, 5 31 3 43 3 43	**iblkigt** **i1igt,i2igt,..j..,k..**	**ibkigt(no),i1igt(no),i2igt(no),j1igt(no),j2igt(no), k1igt(no),k2igt(no),no=1,noigt:** Specification of the position of the ignition spark(s). **ibkigt(no):** The no*th* ignition is located in **ibkigt(no)** block. **i1igt(no):** The starting *i*-index of the ignition spark location. **i2igt(no):** The ending *i*-index of the ignition spark location. **j1igt(no):** The starting *j*-index of the ignition spark location. **j2igt(no):** The ending *j*-index of the ignition spark location. **k1igt(no):** The starting *k*-index of the ignition spark location. **k2igt(no):** The ending *k*-index of the ignition spark location.
1.600000e+03	**toff**	The cut-off temperature within the ignition region to turn off the energy sparks.
5.0e+07	**sparkuser**	The strength of ignition spark (J/(m^3-s)). If zero, then the default strength in the code is used.
0	**nsdata**	= 0 The spray data are to be generated in the following inputs. =1 The spray data are to be provided through user-generated file, *spray.dat*. The format for *spray.dat* is described in another section.
1	**ispray**	= 0 Single-phase flow calculation. (no liquid phase equations are solved) = 1 Two-phase flow calculation. If ispray=0 or nsdata=1, skip to the input for **inplot(i)**.
1	**izospr**	Nozzle zone
3 3 22 28 19 25	**ispray1,ispray2,** **jspray1,jspray2,** **kspray1,kspray2**	The spray particles will be issuing from the (i,j,k) points between **ispray1,ispray2** points in *i* direction, between **jspray1,jspray2** points in *j* direction, and between **kspray1,kspray2** points in *k* direction.

TABLE 4. Detailed Layout of ***allspd.dat*** File

Input Data	Input Variable	Comments
3 45.0 65.0	**nangles, angle1, angle2**	**nangle** = Number of angles between (**angle1,angle2**) from which the spray particles are issuing. **angle1** = The lower angle (say 10 degree) for the spray particles to shoot out. (The angle currently is only in the *x-z* plane). **angle2** = The upper angle (say 45 degree) for the spray particles to shoot out. (The angle currently is only in the *x-z* plane).
5.5 7.5	**size1, size2**	**size1** = The smallest size of spray particles (in terms of μ *m* in radius). **size2** = The largest size of spray particles (in terms of μ *m* in radius).
100	**ntspr**	The time step to initiate the spray subroutine.
20	**ncalspr**	The spray subroutine is called every **ncalspr** flow iterations (pseudo time steps)
0	**imspr**	Always set to 0 (reserved for later version use)
300.0	**tspr**	Liquid spray temperature(K)
4.67e-4	**flowispry**	Liquid spray mass flow rate (kg/s)
50.0	**vispr**	Liquid spray velocity In spray calculations, the number of groups of particles is specified through the parameter statement(**allspd.h**) when compiling. From the above specifications, a complete spray table is established. A spray particle is characterized by its position (i,j,k), angle, size and temperature. The mass of each spray particle is then calculated from the **flowispr** divided by the total number of particles (defined in the allspd.h file). Based on the size information for each particle, a finite number of droplets can be calculated in each spray particle. Therefore, when a spray particle is mentioned, it actually represents a group of small droplets that have the same size, velocity, position, and temperature.
1 2 3 4 5 15 16 17 18 19 11 12 13 14 8	**inplot**	Plot file flags specify the flow variables to be written to the plot3d files. There are three files to be written out for plotting and/ or post-processing purposes. The first file contains vector quantities, **allspd.PFV**, and the second and the third files contain all scalar quantities, **allspd.PFS1** and **allspd.PFS2**. In plot3d, five variables are needed in each file for three-dimensional problems. ALLSPD-3D users are asked to specify 15 flow variables for these three output files. The **inplot(i)** has the following valid integer values. =1 density, = 2 u momentum, = 3 v momentum, = 4 w momentum = 5 static temperature, = 6 gage pressure, = 7 static pressure = 8 turbulent $C\mu$, = 9 static enthalpy, = 10 total enthalpy = 11 turbulent kinetic energy, = 12 turbulent dissipation = 13 turbulent eddy viscosity/laminar viscosity = 14 distance to the walls, = i+14 mass fraction of species i
5	**nspr**	Number species to read in Initial Condition

TABLE 4. Detailed Layout of *allspd.dat* File

Input Data	Input Variable	Comments
'CH3OH' 0.000000e+0	**name, Ysi(1)**	Species name and species mass fraction
'O2' 2.330000e-01	**name, Ysi(2)**	Species name and species mass fraction
'CO2' 0.000000e+00	**name, Ysi(3)**	Species name and species mass fraction
'H2O' 0.000000e+00	**name, Ysi(4)**	Species name and species mass fraction
'N2' 7.670000e-01	**name, Ysi(ns)**	Species name and species mass fraction

4.2.4 User Defined Initial Conditions: *uvwt.dat*

When **iuvwt=1 or 2** in *allspd.dat* is specified, the code needs the *uvwt.dat* data file. This file is read by subroutine INIT.

For **iuvwt=1:** uvwt.dat file must be an unformatted file.

```
do n=1,nblk
ii=imaxbl(n)
jj=jmaxbl(n)
kk=kmaxbl(n)
do k=1,kk
do j=1,jj
do i=1,ii
read(9) pp, u, v, w, t, tke, tep,(ys(nn), nn=1, ns)
end do
```

where
nblk = number of blocks
imaxbl(n) = grid points in *i* direction in n*th* block
jmaxbl(n) = grid points in *j* direction in n*th* block
kmaxbl(n) = grid points in *k* direction in n*th* block
pp = gauge pressure (p-pref)
u = x component of velocity
v = y component of velocity
w = z component of velocity
t = static temperature
tke = turbulent kinetic energy
tep = turbulent dissipation rate
(ys(nn), nn=1, ns) = mass fraction for each species

For **iuvwt=2:** uvwt.dat must be a free formatted file, only the inlet face data are defined in this file (say i=1).

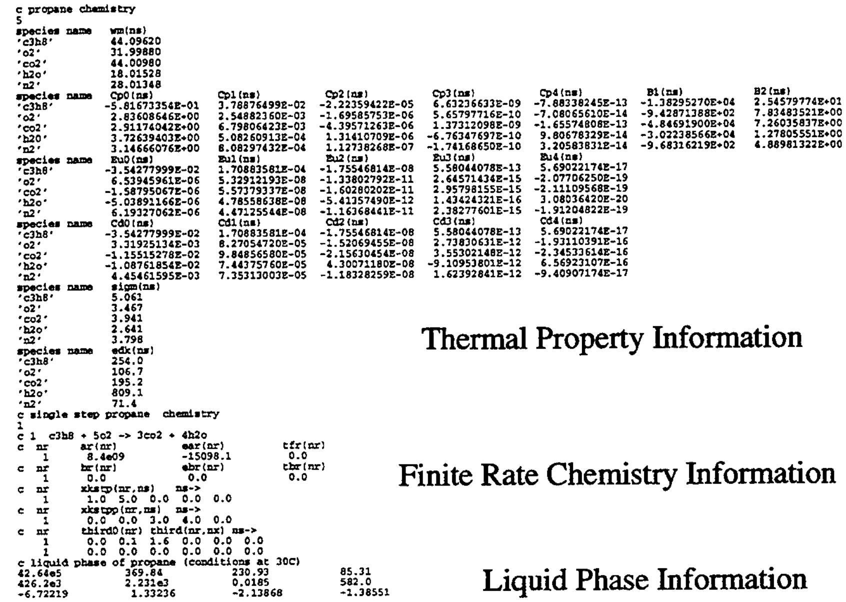
This option is designed for the users who want to specify an inlet swirling profile that can be processed and fitted into the grid points at the inlet face. Right now, a flat inlet face in the west boundary (i=1 and x=constant) is assumed to apply this option. Users have only to provide a radial profile of the swirl cup, and an axisymmetric condition is assumed for the swirling flow at the inlet. If a user's geometry (and grid) does not fit into the above swirling condition, the user can always use the **iuvwt=1** option to define the variables for the entire domain including, of course, the inlet swirling profiles.

```
do n=1,nblk
read(iunit,*) irmax
read(iunit,*) rmax,y0,z0
read(iunit,*) (yscup(nx),nx=1,ns)
do ii=1,irmax
read(iunit,*) rcup(ii),ucup(ii),vcup(ii),wcup(ii), tcup(ii),tkcup(ii)
end do
end do
```

where
nblk = number of blocks
irmax = number of radial points used to define a swirl profile
rmax = radius of the swirler cup (m).
y0 = y coordinate in the center of the swirler cup (m)
z0 = z coordinate in the center of the swirler cup (m)
(yscup(nx),nx=1,ns) = Mass fractions of the species
rcup(ii) = r/rmax, where r is the radial distance from the center of the swirler cup
ucup(ii) = u-velocity (m/s)
vcup(ii) = v-velocity (m/s)
wcup(ii) = w-velocity (m/s)
tcup(ii) = temperature (K)
tkcup(ii) = turbulence intensity

4.2.5 Property Data: *prop.dat*

Property data, chemistry kinetics and liquid fuel property are read by subroutine PROPIN and the data file name is *prop.dat.*



```
c propane chemistry
5
species name   wm(ns)
'c3h8'         44.09620
'o2'           31.99880
'co2'          44.00980
'h2o'          18.01528
'n2'           28.01348
species name   Cp0(ns)          Cp1(ns)          Cp2(ns)          Cp3(ns)          Cp4(ns)          B1(ns)           B2(ns)
'c3h8'        -5.81673354E-01   3.78876499E-02  -2.22359422E-05   6.63236633E-09  -7.88338245E-13  -1.38295270E+04   2.54579774E+01
'o2'           2.83608646E+00   2.54882360E-03  -1.69585753E-06   5.65797716E-10  -7.08065610E-14  -9.42871388E+02   7.83483521E+00
'co2'          2.91174042E+00   6.79806423E-03  -4.39571263E-06   1.37312098E-09  -1.65574808E-13  -4.84691900E+04   7.26035837E+00
'h20'          3.72639403E+00   5.08260913E-04   1.31410709E-06  -6.76347697E-10   9.80678329E-14  -3.02238566E+04   1.27805551E+00
'n2'           3.14666076E+00   8.08297432E-04   1.12738268E-07  -1.74168650E-10   3.20583831E-14  -9.68316219E+02   4.88981322E+00
species name   Eu0(ns)          Eu1(ns)          Eu2(ns)          Eu3(ns)          Eu4(ns)
'c3h8'        -3.54277999E-02   1.70883581E-04  -1.75546814E-08   5.58044078E-13   5.69022174E-17
'o2'           6.53945961E-06   5.32912193E-08  -1.33802792E-11   2.64571434E-15  -2.07706250E-19
'co2'         -1.58795067E-06   5.57379337E-08  -1.60280202E-11   2.95798155E-15  -2.11109568E-19
'h2o'         -5.03891166E-06   4.78558638E-08  -5.41357490E-12   1.43424321E-16   3.08036420E-20
'n2'           6.19327062E-06   4.47125544E-08  -1.16368441E-11   2.38277601E-15  -1.91204822E-19
species name   Cd0(ns)          Cd1(ns)          Cd2(ns)          Cd3(ns)          Cd4(ns)
'c3h8'        -3.54277999E-02   1.70883581E-04  -1.75546814E-08   5.58044078E-13   5.69022174E-17
'o2'           3.31925134E-03   8.27054720E-05  -1.52069455E-08   2.73830631E-12  -1.93110391E-16
'co2'         -1.15515278E-02   9.84856580E-05  -2.15630454E-08   3.55302148E-12  -2.34533614E-16
'h2o'         -1.08761854E-02   7.44375760E-05   4.30071180E-08  -9.10953801E-12   6.56923107E-16
'n2'           4.45461595E-03   7.35313003E-05  -1.18328259E-08   1.62392841E-12  -9.40907174E-17
species name   sigm(ns)
'c3h8'         5.061
'o2'           3.467
'co2'          3.941
'h2o'          2.641
'n2'           3.798
species name   edk(ns)
'c3h8'         254.0
'o2'           106.7
'co2'          195.2
'h2o'          809.1
'n2'           71.4
c single step propane  chemistry
1
c 1  c3h8 + 5o2 -> 3co2 + 4h2o
c  nr      ar(nr)            ear(nr)          tfr(nr)
    1       8.4e09           -15098.1          0.0
c  nr      br(nr)            ebr(nr)          tbr(nr)
    1       0.0               0.0              0.0
c  nr      xkstp(nr,ns)    ns->
    1       1.0  5.0  0.0  0.0  0.0
c  nr      xkstpp(nr,ns)   ns->
    1       0.0  0.0  3.0  4.0  0.0
c  nr      third0(nr) third(nr,nx) ns->
    1       0.0  0.1  1.6  0.0  0.0  0.0
    1       0.0  0.0  0.0  0.0  0.0  0.0
c liquid phase of propane (conditions at 30C)
42.64e5         369.84          230.93          85.31
426.2e3         2.231e3         0.0185          582.0
-6.72219         1.33236        -2.13868        -1.38551
```

Thermal Property Information

Finite Rate Chemistry Information

Liquid Phase Information

FIGURE 6. Typical *prop.dat* File

TABLE 5. Detailed Information about *prop.dat* File

Input Variable	Comments
heading:	title of this file (up to 80 characters)
hsf:	number of species
heading:	title of the property (up to 80 characters)
specnm(nx),wm(nx),nx=1,ns:	specnm(nx) = species name (up to 80 characters). wm(nx) = molecular weight for each species.
heading:	title of the property (up to 80 characters)

TABLE 5. Detailed Information about ***prop.dat*** File

Input Variable	Comments
specnm(nx),cp0(nx),cp1(nx),cp2(nx),cp3(nx),cp4(nx), cp5(nx),cp6(nx)	coefficients for 4th order specific heat polynomial curve as function of temperature The Cp curve is generated by a forth order polynomial defined in McBride[16]. The specific heat, enthalpy and entropy are defined in Equation (30) and Equation (49) where **cp5(nx)** is a constant which combines the heat of formation of each species at 298.15 K and an integration constant for the enthalpy and **cp6(nx)** is an integration constant for the entropy. specnm(nx)=species name (up to 80 characters).
heading:	title of the property (up to 80 characters)
specnm(nx),eu0(nx),eu1(nx),eu2(nx),eu3(nx),eu4(nx)	coefficients for 4th order dynamic viscosity polynomial curve function of temperature
heading	title of the property (up to 80 characters)
specnm(nx),cd0(nx),cd1(nx),cd2(nx),cd3(nx),cd4(nx)	coefficients for 4th order thermal conductivity polynomial curve function of temperature, Equation (41)
heading	title of the property (up to 80 characters)
specnm(nx),sigm (nx)	species name, effective molecular collision area, Equation (45)
heading	title of the property (up to 80 characters)
specnm(nx),edk(nx)	species name, effective temperature(normalized depth of potential energy for a species), Equation (45)
****Kinetic Information*****	See Section 4.4
heading	title of the property (up to 80 characters)
pcri,tcri,tboil,tmelt	Liquid phase critical property information **pcri**-liquid phase critical pressure (N/m^2) **tcri** - liquid phase critical temperature (K) **tboil**- liquid phase boiling point temperature (K) **tmelt**-liquid phase melting point temperature (K)
hfg,cppar,rkpar,rhoipar	**hfg**- latent heat of vaporization (J/kg) **cppar**- liquid phase specific heat (J/kg-K) **rkpar**-liquid phase conductivity (J/m-s-K) **rhoipar**-liquid phase density (kg/m^3)
vp1,vp2,vp3,vp4	Critical pressure curve fit coefficients, Equation (18)

4.2.6 User Defined Spray Data: *spray.dat*

When **nsdata**=1 and **ispray**=1 are specified, the code expects users to supply the spray information from the file, *spray.dat*. The format is described as follows.

```
read(iunit,*) npgrp
do np=1,npgrp
read(iunit,*)
xpar0(np,1),xpar0(np,2),xpar0(np,3),
velpar0(np,1),velpar0(np,2),velpar0(np,3),
tspr0(np),radpar0(np),numpar(np)
end do
```

where
npgrp = number of group of particles for spray
xpar0(np,1) = x coordinate of the initial spray location (m)
xpar0(np,2) = y coordinate of the initial spray location (m)
xpar0(np,3) = z coordinate of the initial spray location (m)
velpar0(np,1) = x component of the spray injection velocity (m/s)
velpar0(np,2) = y component of the spray injection velocity (m/s)
velpar0(np,3) = z component of the spray injection velocity (m/s)
tspr0(np) = liquid temperature (K) for each group of spray
radpar0(np) = radius of each drop size (m)
numpar(np) = number of droplets per spray group (#/s)

4.3 Output Data Files

The ALLSPD-3D code generates up to eleven output data files. The input data is echoed out to *out.dat*. The files, *allspd.PFV, allspd.PFS1,* and *allspd.PFS2* contain solutions in PLOT3D format. If spray was used in the calculation, the spray data files *loss.dat, allspd.SPY* are also generated. The restart information is located in the *save.dat* file. The convergence histories are written out to the *res.dat* and *reske.dat* files. In addition, the turbulence model restart information is written to *the face.dat* file and the pre-processed geometry information is written to *the ynbl.dat* file.

4.3.1 Plot3d Data Files

There are three plot3d files. One is the vector file, *allspd.PFV,* and the other two are scalar files, *allspd.PFS1* and *allspd.PFS2*. The contents for each file depends on the user inputs specified in *allspd.dat*

```
read(iunit,*)nblk
read(iunit,*)(imax(n),jmax(n),kmax(n),n=1,nblk)
do n=1,nblk
read(iunit,*)fm0,angle,reno,time
read(iunit,*)
(((q1(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((q2(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((q3(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((q4(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((q5(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n))
end do
```

where
fm0 = the free stream Mach number
angle = flow angle
reno = flow Reynolds number
time = time
q1 to q5 = are the flow variables defined in the *allspd.dat*

4.3.2 Spray Data Files

All spray files are generated and read within the subroutine SPRAY. The spray results are written into two files. The file *count.dat* contains a brief summary of the initial distribution of the spray particles groups. The file *loss.dat* contains a summary of the spray group time integration history. This file also contains the number of time integrations required for a fuel droplet to vaporize or to hit a particular boundary. The ALLSPD-3D spray model assumes that complete vaporization occurs when a drop hits a wall. It is also assumed that, when a droplet passes through the exit boundary, it is no longer part of the calculation(i.e. particle mass is lost). The current code does not continue to track the particle trajectory once it passes through the interface between blocks (if more than one block is used in the calculation). A true multi-block spray code will be released in the future. Fortunately, in many practical spray applications, the liquid particles evaporate rather quickly before hitting any interface boundary. The maximum time integration limit written out to the *loss.dat* file should be used to set the integration limit in the parameter statement, *allspd.h*, when compiling. Users can also try to increase *npmax* to a large number. The spray data are written into *allspd.SPY* and its format is as follows.

```
write(iunit,*) npgrp
do np=1,npgrp
write(iunit,*) np,npmark(np)
write(iunit,*)
xplot(npidx,1),xplot(npidx,2),xplot(npidx,3),
rplot(npidx),tplot(npidx),dtplot(npidx),npidx=1,npmark(np)
end do
```

where
npgrp = number of group of particles for spray
npmark(np) = the last number of time integration for each group of particle
xplot(npidx,1) = x-coordinate of spray trajectory (m)
xplot(npidx,2) = y-coordinate of spray trajectory (m)
xplot(npidx,3) = z-coordinate of spray trajectory (m)
rplot(npidx) = radius of the particle along its trajectory
tplot(npidx) = droplet surface temperature along its trajectory (K)
dtplot(npidx) = integration time step along its trajectory (s)

4.3.3 Restart File

The primary restart file is called *save.dat*. This file, and the files *face.dat* and *ynbl.dat*, are the only files required to restart a previous ALLSPD-3D run. They contain all necessary data to restart the program. It should be noted that the *save.dat* files generated by serial mode and parallel mode execution are not compatible. Therefore, users should not try to restart a run in serial mode with the restart file that is generated by a parallel version, and *vice versa*. The file *save.dat* is generated in subroutine SAVE and is read by subroutine RESTART. The file *face.dat* is generated in subroutine SAVE and is read in by subroutine INIT. The file *ynbl.dat* is generated in subroutine PREGRID and is read in by subroutine MESH.

4.3.4 Residual File

The code writes the convergence history to both the computer screen and files named *res.dat* and *reske.dat*. The residual shown on the computer screen is the

L2 norm of the total residual for flow variables and the residuals shown in the *res.dat* contain residual for different flow variables. The write statement is *write(iunit,*) itertot, rescon, res_p, res_u, res_v, res_w, res_h* where

itertot: iteration number
rescon: L2 norm of residual for flow variables
res_p: residual for pressure
res_u: residual for u velocity
res_v: residual for v velocity
res_w: residual for w velocity
res_h: residual for energy
The *reske.dat* contains residual for turbulent variables, i.e., κ and ε.

4.4 Typical Chemistry Input File Structure

The most important rule to chemistry inputs is that N_2 must be the last species in the input file. For turbulent chemistry model calculations, the first species and second species must be fuel and air, respectively.

4.4.1 One-Step Global Chemistry Mode

TABLE 6. Typical Single-step Chemistry Input File

Input file content	**Comments**
c methanol chemistry	Title
1	Number of reactions
c nr ar(nr) ear(nr) tfrr	Heading Title for Forward reaction rates
1 4.6e11 -15098.1 0.0	Reaction number, Pre-exponent, Activation Temperature[K], Temperature Correction Factor
c nr br(nr) ebr(nr) tbrr	Heading Title for Backward reactions
c 1 ch3oh + 1.5o2 -> co2 + 2h2o	Title, helpful to write the reaction mechanism (blank lines are acceptable)
1 0.0 0.0	Reaction number, Pre-exponent, Activation Temperature[K], Temperature Correction Factor (note this is forward only reaction and GUI will not generate this line!)
c nr xkstp(nr,ns) ns->	Heading, Forward Stoichiometric Coefficients
1 1 1.5 0 0 0	Reaction number, Forward Stoichiometric Coefficients (There should be nspe of them)
c nr xkstpp(nr,ns) ns->	Heading, Backward Stoichiometric Coefficients
1 0 0 1 2 0	Reaction number, Backward Stoichiometric Coefficients (There should be nspe of them)
c nr third0(nr) third(nr,ns) ns->	Heading Title for Correction Mechanism (Note, this input is to used to change the power of the reaction constant)
1 -1.0 1.0 1.0 0.0 0.0 0.0	Reaction number, Reaction type, Power of the reaction mechanism in forward direction (for the nspe species)
1 -1.0 0.0 0.0 0.0 0.0 0.0	Reaction number, Reaction type, Power of the reaction mechanism in Reverse direction (for the nspe species)
Note	**Third0(1)** = -1 ,Global type kinetics **Third0(1)** = 0, Modified kinetics with fractional power **Third0(1)** >0 detailed kinetics with 3-body corrections $K_f = A \exp(E_a/RT)\, T^0\, [ch3oh]^1\, [02]^1$ $K_b = 0.0$ Mechanism of Westbrook and Dryer[37]

4.4.2 Simple Two-Step Global Chemistry Model

TABLE 7. Typical Two-step Chemistry Input File

Input file content	Comments
cTwo-step propane chemestry	Title
2	Number of reactions
c 1 c3h8 + 7/2 o2 -> 3co + 4h2o c 2 co + .5o2 <-> co2	Title, helpful to write the reaction mechanism (blank lines are acceptable)
c nr ar(nr) ear(nr) tfrr	Heading Title for Forward reaction rates
1 5.62e09 -15098.1 0.0 2 2.24e12 -20140. 0.0	Reaction number, Pre-exponent, Activation Temperature[K], Temperature Correction
c nr br(nr) ebr(nr) tbrr	Heading Title for Backward reactions
1 0.0e00 0.0 0.0 2 5.00e08 -20140 0.0.	Reaction number, Pre-exponent, Activation Temperature[K], Temperature Correction factor (reaction one is a global reaction and therefore does not have any reverse reactions)
c nr xkstp(nr,ns) ns->	Heading for Forward Stoichiometric Coefficients
1 1 3.5 0 0 0 0 2 0 0.5 1.0 0 0 0	Reaction number, Forward Stoichiometric Coefficients (There should be **nspe** of them)
c nr xkstpp(nr,ns) ns->	Heading for Backward Stoichiometric Coefficients
1 0 0 3 0 4 0 2 0 0 0 1.0 0 0	Reaction number, Backward Stoichiometric Coefficients (There should be **nspe** of them)
c nr third0(nr) third(nr,ns) ns->	Heading Title for Correction Mechanism (This input is used to change the power of the reaction constant)

TABLE 7. Typical Two-step Chemistry Input File

Input file content	**Comments**

1 0.0 0.1 1.65 0.0 0.0 0.0 0
1 0.0 0.0 0.0 0.0 0.0 0.0 0
2 0.0 0.0 0.25 1.0 0.0 0.5 0
2 0.0 0.0 0.0 0.0 1.0 0.0 0
	Reaction number, Reaction type, Power of the reaction mechanism in forward and backward direction (Note that this mechanism required modified power for both reactions)

NOTE:	Third0(1) = -1 ,Global type kinetics Third0(1) = 0, Modified kinetics with fractional power Third0(1) >0 detailed kinetics with 3-body corrections c3h8 + 7/2 o2 -> 3co + 4h2o $K_f = A \exp(E_a/RT)\ T^0\ [c_3h_8]^{0.1}\ [0_2]^{1.65}$ $K_b = 0.0$ co + .5o2 <-> co2 $K_f = A \exp(E_a/RT)\ T^0\ [CO]^1\ [H_2O]^{0.5}\ [O_2]^{0.25}$ $K_b = A \exp(E_b/RT)\ T^0\ [CO_2]^1$ Mechanism of Westbrook and Dryer[37]

4.4.3 Typical Detailed Chemistry Input file

TABLE 8. Detailed Chemistry Input File

Input file content	Comments
c Large complex chemistry	Title
18	Number of reactions
c1 H_2 + OH <-> H_2O + H c2 O_2 + H < -> OH + O c3 H_2 + O <-> OH + H c4 O_2 + H <-> HO_2 c5 H_2 + M < -> H + H c6 O_2 + M <-> O + O c7 H + HO_2 < -> H_2 + O c8 H_2O + M <-> OH + H c9 H_2O + O <-> 2OH c10 O + HO_2 <-> O_2 + OH c11 H + HO_2 <-> 2OH c12 OH + HO_2 < -> O + H_2O c13 H_2 + H_2O_2 <-> H_2O + HO_2 c14 OH + H_2O_2 <-> H_2O + H_2O_2 c15 2HO_2 + M <-> O2 + H_2O_2 c16 H + H_2O_2 <-> OH + H_2O c17 H_2O_2 +M <-> 2OH c18 H + O <-> OH+ M	Title, helpful to write the reaction mechanism (blank lines are acceptable)
c nr ar(nr) ear(nr) tfrr	Heading Title for Forward reaction rates

TABLE 8. Detailed Chemistry Input File

Input file content	Comments
1 4.74E+10 -3068.95 0.0	Reaction number, Pre-exponent, Activation Temperature[K], Temperature Correction
2 1.85E+11 -8253.65 0.0	
3 4.2E+11 -6919.98 0.0	
4 1.35E+9 503.27 0.0	
5 2.2E+11 -48314.04 0.0	
6 1.8E+15 -59396.074 1.0	
7 1.3E+11 -1509.814 0.0	
8 1.3E+12 -52913.94 0.0	
9 6.8E+10 -9242.577 0.0	
10 5.0E+10 -503.27 0.0	
11 2.04E+11 -538.5 0.0	
12 8.0E+9 -1499.75 0.0	
13 7.91E+10 -12581.78 0.0	
14 6.1E+9 -719.678 0.0	
15 1.8E+9 0.0 0.0	
16 7.8E+8 0.0 0.0	
17 1.44E+14 -22903.875 0.0	
18 7.1E+12 0.0 0.0	
c nr br(nr) ebr(nr) tbrr	Heading Backward reactions
1 2.03138E+12 -10985.23 0.274934	Reaction number, Pre-exponent, Activation Temperature[K], Temperature Correction
2 4.52315E+08 412.94 0.410439	
3 2.2825E+11 -5998.66 0.0246801	
4 5.1015E+13 -25765.13 0.423663	
5 4.3848E+7 4166.86 0.0320618	
6 1.51713E+9 840.326 0.525515	
7 1.275E+10 -27672.1 0.4273321	
8 5.73655E+6 7491.96 0.313192	
9 8.613E+8 -404.787 0.250384	
10 2.6504E+9 -25743.37 0.403293	
11 2.768E+7 -17119.5 0.8083150	
12 4.05365E+10 -35609.15 0.130335	
13 5.07334E+12 -3160.89 0.411882	
14 3.761E+09 -18043.58 0.14644	
15 1.0845E+10 -16734.1 0.0205195	
16 1.04212E+5 -33982.3 0.89952	
17 8.487E+4 3519.125 0.0	
18 2.1057E+16 -51573.5 1.0666587	
c nr xkstp(nr,ns) ns->	Heading

TABLE 8. Detailed Chemistry Input File

Input file content	Comments
1 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0	Reaction number, Forward Stoichiometric Coefficients (There should be **nspe** of them)
2 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0	
3 1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0	
4 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0	
5 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0	
6 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0	
7 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0	
8 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0	
9 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0	
10 0.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0	
11 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0	
12 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0	
13 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0	
14 0.0 0.0 1.0 0.0 0.0 0.0 0.0 1.0 0.0	
15 0.0 0.0 0.0 0.0 0.0 0.0 2.0 0.0 0.0	
16 0.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0	
17 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0	
18 0.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0	
c nr xkstpp(nr,ns) ns->	Heading
1 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0	Reaction number, Backward Stoichiometric Coefficients (There should be **nspe** of them)
2 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0	
3 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0	
4 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0	
5 0.0 0.0 0.0 0.0 2.0 0.0 0.0 0.0 0.0	
6 0.0 0.0 0.0 0.0 0.0 2.0 0.0 0.0 0.0	
7 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0	
8 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0	
9 0.0 0.0 2.0 0.0 0.0 0.0 0.0 0.0 0.0	
10 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0	
11 0.0 0.0 2.0 0.0 0.0 0.0 0.0 0.0 0.0	
12 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0	
13 0.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0	
14 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0	
15 0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0	
16 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0	
17 0.0 0.0 2.0 0.0 0.0 0.0 0.0 0.0 0.0	
18 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0	

TABLE 8. Detailed Chemistry Input File

Input file content	Comments
c nr third0(nr) third(nr,nx) ns->	Heading

1 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
2 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
3 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
4 2.0 2.5 0.3 0.0 20.3 0.0 0.0 0.0 0.0 0.0
5 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
6 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
7 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
8 2.0 3.2 3.2 15.7 0.0 0.0 0.0 0.0 0.0 0.0
9 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
10 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
11 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
12 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
13 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
14 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
15 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
16 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
17 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
18 2.0 4.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
	Reaction number, Reaction type, Power of the reaction mechanism in forward direction (for the nspe species)

	Taken from Lee[35]

V : Graphical User Interface(GUI)

This is a preliminary manual for alpha version 0.1 of the ALLSPD-3D Graphical User Interface (GUI). This GUI corresponds to version 1.0 of the ALLSPD-3D Code.

5.1 Installation

The ALLSPD-3D GUI was written in C and requires X/Motif Release 1.1. It has been built successfully on the following platforms: IBM RS6000, SGI, Also beta ported to Solaris and HPUX.

To install the GUI, first create a separate directory and untar the GUI tar file. This will create a directory tree with the following structure:

```
CHANGES       chemistry/    help_files/   species/
Makefile      common/       include/      view/
Manual/       file/         model/        xmallspd.c
bin/          grid/         solve/
```

Set the XMALLSPD_ROOT environment variable to point to the directory where the GUI will be built. For instance:

setenv XMALLSPD_ROOT /u/quealy/gui

Also, modify your path to include $XMALLSPD_ROOT and $XMALLSPD_ROOT/bin. For example:

set path=($path $XMALLSPD_ROOT $XMALLSPD_ROOT/bin)

These can be included in your .cshrc file so they are issued every time a new shell is created.

A top level Makefile will compile the entire directory structure and create a GUI executable named *'xmallspd'*. At this point, you can type *'make'*, and you should see the code compile. Sample output from this stage is included below.

```
>make
        dir_list="file  grid  model  solve  view  common" ; \
        for dir in $dir_list ; do \
                echo making GUI in directory $dir ; \
                ( cd $dir ; make all ) ; \
        done
making GUI in directory file
        cc -g -I/usr/lerc/fsang/gui/gui/include -c file.c
        echo 'FILE module built...'
FILE module built...
making GUI in directory grid
        cc -g -I/usr/lerc/fsang/gui/gui/include -c grid.c
        echo 'GRID module built...'
GRID module built...
making GUI in directory model
        cc -g -I/usr/lerc/fsang/gui/gui/include -c model.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c geom_info.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c geom_scale.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c geom_intblk.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c geom_file.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c geom_Pfile.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c geom_Cfile.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c geom_util.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c problem.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c prop_fluid.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c prop_spray.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c prop_chem.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c prop_gas_species.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c prop_reaction.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c prop_util.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c prop_ig.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c bc.c
        echo 'MODEL module built...'
MODEL module built...
making GUI in directory solve
        cc -g -I/usr/lerc/fsang/gui/gui/include -c solve.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c control.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c output.c
        cc -g -I/usr/lerc/fsang/gui/gui/include -c solution.c
        echo 'SOLVE module built...'
SOLVE module built...
making GUI in directory view
        cc -g -I/usr/lerc/fsang/gui/gui/include -c view.c
        echo 'VIEW module built...'
VIEW module built...
making GUI in directory common
        cc -g -I/usr/lerc/fsang/gui/gui/include -c util.c
        echo 'COMMON module built...'
COMMON module built...
        make xmallspd
        cc -g -I/usr/lerc/fsang/gui/gui/include -c xmallspd.c
        cc -o xmallspd xmallspd.o file/file.o grid/grid.o model/model.o
                model/geom_file.o model/geom_Pfile.o model/geom_Cfile.o
                model/geom_info.o model/geom_scale.o model/geom_intblk.o
                model/geom_util.o model/problem.o model/prop_fluid.o
                model/prop_spray.o model/prop_chem.o model/prop_gas_species.o
                model/prop_reaction.o model/prop_util.o model/prop_ig.o
                model/bc.o solve/solve.o solve/control.o solve/output.o
                solve/solution.o view/view.o common/util.o -lXm -lXt -lX11 -lPW
```

At the time of writing this manual, several assumptions have been made, especially pertaining to running the parallel version of the code.

Assumptions:

If PVMis used, it is installed and the proper PVM environment variables have been set.
Both the GUI server and all compute nodes share a common file system.
User ids are identical across all platforms.
If LSF is used, it is available on the GUI host as well as the compute nodes.
Pre and post processors exist on the host where the GUI is running, and all executables are in the user's path.
If no reactions are specified in a *prop.dat* file, it is assumed that there are no liquid properties.

The GUI sets default values for most variables used by the ALLSPD-3D code. These default values are set in two locations in the GUI code:

model/model.c: MODEL_InvokeModelSimDefaults ()

solve/solve.c: SOLVE_InvokeSolveSimDefaults ()

The user may edit these routines and adjust the default values if desired, and then recompile the GUI to generate a new *'xmallspd'* executable.

5.2 General Overview

The ALLSPD-3D GUI allows the user to create the following ALLSPD-3D input files:

allspd.dat	(Input file)
prop.dat	(Property file)
gridlink.dat	(Grid linkage file)

The GUI allows a user to start either the serial or parallel code on local or remote hosts, using several startup options. This is subject to the assumptions listed in the installation section.

The ALLSPD-3D GUI also gives a user access to a number of pre- and post-processing tools, again subject to the assumptions in the installation section.

The following 'Scorecard' is included for people who wish to work with the GUI source code itself. This is a cross reference between variable names used by the ALLSPD-3D code, and their corresponding GUI variable names.

Allspd3d	GUI
iread	_solve.control.use_restart
ntmax	_solve.control.iteration.iterations
nplot	_solve.control.iteration.iter_plot
nsave	_solve.control.iteration.iter_save
iturb	_model.prob.turbulence
icmusw	_model.prob.variable_cmu
idisbc	_model.prob.epsilon_bc
icomb	_model.prob.combustion
iartv	_solve.control.misc.artificial_visc
vis4	_solve.control.misc.coeff4
vis2	_solve.control.misc.coeff2
cfl(1)	_solve.control.misc.cfl.eq1
cfl(2)	_solve.control.misc.cfl.eq2
cfl(3)	_solve.control.misc.cfl.eq3
precfac(1)	_solve.control.misc.precondition.eq1
precfac(2)	_solve.control.misc.precondition.eq2
precfac(3)	_solve.control.misc.precondition.eq3
nswitch	_solve.control.iteration.iter_newton
facini	_solve.control.iteration.iter_factor
nkecall	_solve.control.iteration.iter_ke
nscall0	_solve.control.iteration.iter_species
iuvwt	_model.prop.fluid.input_src
uin	_model.prop.fluid.init_vel_x
vin	_model.prop.fluid.init_vel_y
win	_model.prop.fluid.init_vel_z
tref	_model.prop.fluid.init_temp
pref	_model.prop.fluid.init_pressure
rlref	_model.prop.fluid.ref_length
iebm	_model.prob.eddy_breakup
ieqrate	_model.prop.chem.reaction.equilibrium_rate
nt1igt	_model.prop.ignition.ig_start
nt2igt	_model.prop.ignition.ig_end
noigt	_model.prop.ignition.num_srcs;
ibkigt	_model.prop.ignition.ignition_src->block_no
i1igt	_model.prop.ignition.ignition_src->istart
i2igt	_model.prop.ignition.ignition_src->iend
j1igt	_model.prop.ignition.ignition_src->jstart
j2igt	_model.prop.ignition.ignition_src->jend
k1igt	_model.prop.ignition.ignition_src->kstart
k2igt	_model.prop.ignition.ignition_src->kend
toff	_model.prop.ignition.temp_limit
sparkuser	_model.prop.ignition.spark_strength
nsdata	_model.prop.spray.input_src
ispray	_model.prob.spray
izospr	_model.prop.spray.nozzle_block
ispray1	_model.prop.spray.location->istart
ispray2	_model.prop.spray.location->iend
jspray1	_model.prop.spray.location->jstart
jspray2	_model.prop.spray.location->jend
kspray1	_model.prop.spray.location->kstart
kspray2	_model.prop.spray.location->kend
nangles	_model.prop.spray.num_angles
angle1	_model.prop.spray.angle_start
angle2	_model.prop.spray.angle_end
size1	_model.prop.spray.particle_size_start
size2	_model.prop.spray.particle_size_end
ntspr	_model.prop.spray.spray_start
ncalspr	_model.prop.spray.spray_freq
impspr	_model.prop.spray.source_term
tispr	_model.prop.spray.temp
flowispr	_model.prop.spray.initial_prop.input_data.mass_flow_rate
vispri	_model.prop.spray.initial_prop.input_data.vel_mag
inplot	_solve.output.plot3d
iblank	_model.geom.iblank

The main menu of the ALLSPD-3D GUI is illustrated in Figure 7 . The remaining sections of this chapter highlight each of the options from this main menu briefly describes each function.

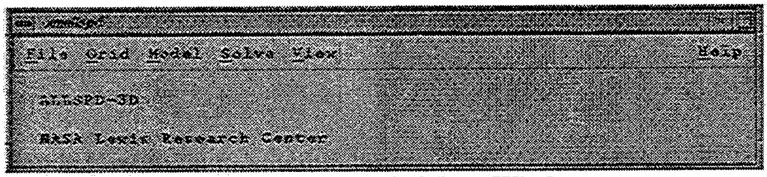


FIGURE 7. **ALLSPD-3D GUI Main Menu Window**

5.3 File

The File section of the main menu allows you to read and write the three main ALLSPD-3D files (input, property and grid linkage files), and also to exit the GUI.

For clarification in this manual, the files will be referred to by their default names: *allspd.dat* (input file), *gridlink.dat* (grid linkage file), and *prop.dat* (property file). Note that the GUI accepts and will generate filenames which differ from the default. The default names are used in this manual for simplicity.

The order of reading the three input files can be important. Since species information may be included in each of these three files, it is wise to read an existing *prop.dat* file first. If *prop.dat* is not read first, all species information at the end of the *allspd.dat* file and the *gridlink.dat* file is ignored, and a warning message is displayed.

When reading the *gridlink.dat* file, the *grid.dat* filename is also requested. The GUI does not read the entire *grid.dat* file. It simply reads the initial information in the file to verify that the *gridlink.dat* file matches the *grid.dat* file.

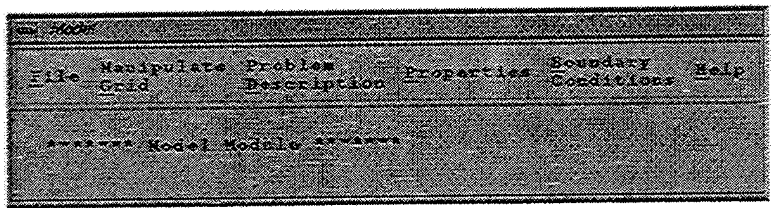
5.4 Grid

The Grid section of the main menu gives the user access to grid generation packages. Currently the GUI allows starting the CFD-GEOM package developed by CFD Research. Gridgen access has not yet been implemented.

The user may set the XMALLSPD_CFDGEOM_EXEC environment variable to point to the location of the CFD-GEOM executable. If this environment variable is not set it is assumed the default name 'CFD-GEOM' is used, and that the executable is in the user's path. It is assumed that licenses to access CFD-GEOM are in place, and that CFD-GEOM will run on the same host as the GUI.

5.5 Model

The model menu is illustrated in Figure 8 . Each of the options from this model menu are briefly described below



******* Model Module *******

FIGURE 8. ALLSPD-3D GUI Model Menu Window

5.5.1 Model/File

The File option from the Model Module simply allows you to close the Model Module window.

5.5.2 Model/Manipulate Grid

The 'Manipulate Grid' option from the Model Module window allows the user to import a grid developed externally, display grid information, perform grid scaling if necessary, and define internal blockages.

The GUI currently supports PLOT3D and CFD-GEOM grid files. The files can be 2D or 3D files. All 2D files are converted by the GUI to a 3D format usable by ALLSPD-3D. The option to import a Gridgen grid file has not been implemented yet.

The 'Grid Information' option displays grid parameter information for each grid block. If a grid file has not been specified, and error message is displayed.

The 'Grid Scaling' option allows the user to scale the current grid from either inches, feet or centimeters to meters (which is what ALLSPD-3D expects). The user is allowed the option to select a new file name. If a new file name is not selected, the original grid is overwritten.

WARNING: Since this is an alpha version of the GUI, it is STRONGLY ADVISED that you do not overwrite the original grid. Any errors occurring during the scaling process could destroy the original file. To prevent this from happening, keep a copy of the original files in a protected place with read only access.

The 'Define Internal Blockages' option allows the user to define any number of internal blockages for each block.

5.5.3 Model/Problem Description

The 'Problem Description' section of the Model Module allows the user to select several problem options. The options to select are:

Turbulent
Reacting
Spray
Eddy Breakup Model

If the application is Turbulent, the user has the additional option of using a variable $C\mu$ in the turbulent eddy viscosity formulation or the CMOTT ε boundary condition in place of the default ε boundary condition.

5.5.4 Model/Properties

The ‘Properties’ section of the Model Module is the most complex section of the ALLSPD-3D GUI. Under this option, the user may specify Fluid properties, Spray properties, Chemistry properties (Thermal & Kinetics), and Ignition properties. Each of these areas is described in detail below.

5.5.4.1 Model/Properties/Fluid

Under the Fluid Properties window, the user is able to specify various initial fluid properties, including:

Initial Velocity (x,y,z)
Initial Temperature
Initial Pressure
Reference length

Also, the user is given four options for setting initial flow conditions:

Initial velocity plane is defined ONLY
Uniform Initial conditions
User Defined Initial conditions (expects uvwt file)
Initial plane from Swirl Cups (expects uvwt file)

At the time of writing this manual the *uvwt.dat* filename must be entered via the Model/Properties/Fluid window if the *allspd.dat* file indicates that such a file is to be read. If the ALLSPD-3D code expects the filename but it has not been entered through the GUI, an error message is displayed when the user attempts to start the code.

5.5.4.2 General Information on Species Database

The GUI relies on a species database to obtain property information for the *prop.dat* file. The species database that is distributed with the GUI source code is located in *$XMALLSPD_ROOT/species/species_list.* There are currently 31 species in this database.

The user is able to customize this species database by entering information via the GUI and storing the newly created species information into the database. This allows the user to add to this database as desired.

WARNING: Since this is an alpha version of the GUI, it is STRONGLY ADVISED that you keep a backup copy of any species database that is built by the user. Any errors occurring during the writing of new information to the existing species database could destroy the existing database. To prevent this from happening, keep a copy of a customized species database in a protected place with read only access. Also, if you are in the process of creating a species database, save it frequently and check for errors. And do not forget to make backups!

5.5.4.3 Locating the Master Species Database

The GUI follows a pre-defined search path when trying to locate the Master Species Database. First the environment variable XMALLSPD_SPECIES_LIST is examined. If this environment variable has been defined, the species database is read from this location. If not defined, the search path continues in the following order:

.xmallspd/species_list

$HOME/.xmallspd/species_list

$XMALLSPD_ROOT/species/species_list

The first species database file which is located is then used to read the master species list into the database.

When updating the master list, the file search order is as follows:

$XMALLSPD_SPECIES_LIST

.xmallspd/species_list

$HOME/.xmallspd/species_list

If the file is not located in any of these locations, this means that the original distribution species list has been altered. To prevent overwriting the original, the *.xmallspd* directory is created in the current working directory, and the species list is written there *(.xmallspd/species_list).*

The GUI uses the chemical name for searching the database, so this must always be unique. Also, at the time of writing this manual, the GUI is 'case sensitive.' Most species in the database have been entered with uppercase letters. Some however require lowercase letters (such as iC4H10 for iso-Butane).

5.5.4.4 Locating the Master 'Default Chemistry Package' Database

The GUI follows a pre-defined search path when trying to locate the Master 'Default Chemistry Package' Database. First the environment variable XMALLSPD_CHEMISTRY_LIST is examined. If this environment variable has been defined, the DefaultChemList (and any selected packages) are read from this location. If not defined, the search path continues in the following order:

.xmallspd/DefaultChemList

$HOME/.xmallspd/DefaultChemList

$XMALLSPD_ROOT/chemistry/DefaultChemList

The first DefaultChemList database file which is located is then used to read the master 'Default Chemistry Package' list into the database.

When updating the master 'Default Chemistry Package' list, the file search order is as follows:

$XMALLSPD_CHEMISTRY_LIST .xmallspd/DefaultChemList

$HOME/.xmallspd/DefaultChemList

If the file is not located in any of these locations, this means that the original distribution DefaultChemList has been altered. To prevent overwriting the original, the *.xmallspd* directory is created in the current working directory, and the DefaultChemList is written there *(.xmallspd/DefaultChemList)*.

5.5.4.5 Model/Properties/Spray Initialization

The user cannot enter the Spray Initialization window unless the Spray problem option has been set. Once this window is activated, the user may either enter spray initialization information (default), or read this information from a file (*spray.dat*).

At the time of writing this manual the *spray.dat* filename must be entered via the Model/Properties/Spray window if the *allspd.dat* file indicates that such a file is to be read. If the ALLSPD-3D code expects the filename but it has not been entered through the GUI, an error message will be displayed when the user attempts to start the code.

If the user opts to enter the spray initialization information, another window opens allowing the user to enter information such as:

Block with nozzle
Spray location
Number of angles and range
Particle size information
Spray source term treatment option
Iteration to start spray
Spray Frequency
Spray temperature
Mass flow rate
Velocity magnitude

5.5.4.6 Model/Properties/Customize Liquid Species

The user is able to either select the Spray Species from a list, or to customize the spray property information via the GUI. If the user chooses to select the spray species from a list, the list is built from the current species database, as determined above.

If the user opts to customize the spray species information, this information can then be added to the user's species database for future use. Properties to enter for customizing include:

Species Name	(for example, 'Oxygen')
Chemical Name	(for example, 'O2')
Critical Pressure	
Critical Temperature	
Melting Temperature	
Boiling Temperature	
Phase change enthalpy	
Specific heat for liquid phase	
Thermal conductivity for liquid phase	
Density at injection temperature	
Vapor pressure coefficients	

5.5.4.7 Model/Properties/Read Default Chemistry Package

This section of the GUI allows the user to read a pre-defined default chemistry package. Several default chemistry packages have been distributed with the ALLSPD-3D GUI. The following packages currently exist:

Heptane
Hydrogen Air
Methanol
Propane(1step)
Air

By building the species database and customizing the thermal properties and kinetics, the user can continue to build and add to the default chemistry packages.

5.5.4.8 Model/Properties/Write Default Chemistry Package

This section of the GUI allows the user to write a customized chemistry package to the chemistry database. Thermal and Kinetics information must be specified before the user is allowed to generate a chemistry package. Once the new package is written to the database, the user is able to read the default chemistry package just like the pre-defined packages which were supplied with the GUI.

NOTE: Currently if a name is entered for a new Default Chemistry Package which duplicates a pre-existing package, the GUI posts an error message and expects the user to select a new name. More flexibility may be added at a later date.

5.5.4.9 Model/Properties/Customize Thermal

The 'Customize Thermal' option allows the user to either select Thermal Properties from the species database, or to enter new species thermal properties by hand.

If the user chooses to enter the Thermal property information, the following information is requested for the species database:

Species name	(for example, Oxygen)
Chemical name	(for example, 'O2')
Specific Heat	
Enthalpy constant	
Entropy constant	
Viscosity	
Thermal Conductivity	
Diffusion parameters	

There are some restrictions upon the order in which species should be selected. The eddy breakup model assumes that the major fuel constituent is first and that oxygen (O2) is second. This order is also assumed for a global reaction mechanism. In general, if nitrogen is chosen as a component, it should be picked as the last species in the list so that round-off error will be minimized.

5.5.4.10 Model/Properties/Customize Kinetics

The user cannot enter the 'Customize Kinetics' window unless the 'Reacting' problem option has been set. Once this window is activated, the user is able to enter reaction information.

The user may enter a reaction set name. Also, the user must specify the known reaction rates (forward reaction rate constants only, backward reaction rate constants only, or both) for all reactions.

The user may then add any number of reactions. For each reaction, the user is asked for:

Reaction type (Global, Westbrook & Dryer, Finite Rate)
Forward Modified Arrhenius rates
pre-exponential
temperature exponent
activation energy
Backward Modified Arrhenius rates
pre-exponential
temperature exponent
activation energy

And for each species,

Forward Stoichiometric Coefficient
Forward Mass Action Coefficient (if appropriate)
Backward Stoichiometric Coefficient

Backward Mass Action Coefficient (if appropriate)
Third Body Efficiency (if appropriate)

5.5.4.11 Ignition

The 'Ignition' window allows the user to enter various ignition properties, including:

Iteration to apply ignition
Iteration to stop ignition
Temperature limit
Spark strength
Ignition Locations

5.5.4.12 Initial Mass Fractions

Once the user selects species information (via 'Default Chemistry Package' or 'Customize Thermal'), a window requesting mass fraction information is displayed. The user is asked to enter an initial mass fraction value for each species that has been selected.

5.5.5 Model/Boundary Conditions

The 'Boundary Condition' window allows the user to specify boundary conditions for each block on each of six faces. It also allows specifying Dilution Holes and Transpiration Cooling areas.

The boundary condition options include:

Upstream
Downstream
Adiabatic Wall
Isothermal Wall
Block Interface
Symmetry
Periodic

Singularity
Far Field

If a downstream or far field boundary condition is specified, the user is asked for an exit pressure value. If an isothermal wall is specified, the user is asked to enter a wall temperature value. If a block interface is specified, the user is asked to enter the block number of the neighboring block.

If a particular block has one or more dilution holes, the user can enter dilution hole information by pressing the 'Dilution Hole' pushbutton. The user may enter any number of dilution holes. The user is asked to enter the following information for each dilution hole:

Dilution hole location
Mass flow rate
Temperature
Turbulent intensity
Direction cosine (x,y,z)

If species have been selected, the user is asked to enter dilution mass fractions for each species.

If a particular block has one or more Transpiration Cooling areas, the user can enter transpiration cooling information by pressing the 'Transpiration Cooling' pushbutton. The user may enter any number of transpiration cooling areas. The user is asked to enter the following information for each transpiration cooling area:

Transpiration cooling area location
Mass flow rate
Direction cosine (x,y,z)

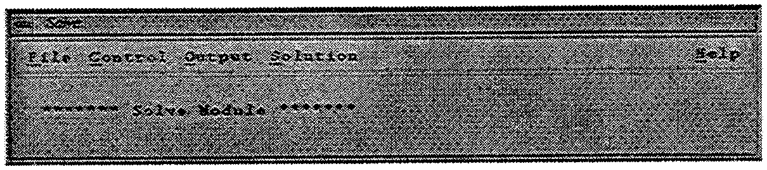
If species have been selected, the user is asked to enter transpiration mass fractions for each species.

5.5.6 Model/Help

Help files are included in the GUI. If a help file does not currently exist, pressing that 'Help' button will have no affect.

5.6 Solve

The solve menu is illustrated in Figure 9 . Each of the options from this solve menu are briefly described below.



******* Solve Module *******

FIGURE 9. ALLSPD-3D GUI Solve menu window

5.6.1 Solve/File

The File option from the Solve Module simply allows you to close the Solve Module window.

5.6.2 Solve/Control

The 'Control' menu option allows the user to specify whether a restart file is to be used, and to set iteration and other miscellaneous parameters.

The 'Restart' option allows the user to specify whether a restart file is to be used. If a restart file is to be used, a file selection box is displayed so the user may enter the name of the restart file. Note that if you are running the parallel code, a restart file exists for each process. These are named:

save.dat-001

save.dat-002

...

save.dat-00n

where n is the number of processes used in this particular calculation. When entering the filename for the restart file in this situation, do not enter a specific file extension. In other words, enter *save.dat* rather than *save.dat-001.*

Also note that a restart file created by the serial code is NOT compatible with the parallel code, and vice versa.

At the time of writing this manual the restart filename must be entered via the Solve/Control window if the *allspd.dat* file indicates that a restart file is to be used. If the ALLSPD-3D code expects the filename but it has not been entered through the GUI, an error message will be displayed when the user attempts to start the code.

The 'Iterations' window allows the user to specify the following parameters:

Total number of Iterations
Iteration frequency for writing a PLOT3D file
Iteration frequency for writing a restart file
Number of iterations between turbulent equation iterations
Number of iterations between solutions to the species continuity equations
Number of iterations to under relax the time step
Relaxation factor for time step

The 'Miscellaneous' window allows the user to specify the following parameters:

Artificial Viscosity
CFL Numbers
Pre-conditioning numbers

If Artificial Viscosity is 'ON', the user is asked for 2nd and 4th order coefficients.

5.6.3 Solve/Output

The 'Output' window allows the user to specify PLOT3D parameters. The user is given a list of variables which can be written to PLOT3D files. The list includes:

Density
U Momentum
V Momentum
W Momentum
Static Temperature
Gage Pressure
Static Pressure
Turbulent CMU
Static Enthalpy
Total Enthalpy
Turbulent Kinetic Energy
Dissipation
Turbulent Eddy Viscosity/Laminar Viscosity
Normal Distance

plus all mass fractions for the selected species.

A maximum of 15 choices can be selected. These are written in the order selected to three PLOT3D files. The first five selections are written to the first PLOT3D file (default = *allspd.PFV*). For compatibility with CFD-VIEW, this file should include any vector quantities, e.g. momentum, as the 2nd, 3rd, and 4th choices. The second five selections are written to the second PLOT3D file (default = *allspd.PFS1*). And the third set of five are written to the final PLOT3D file (default= *allspd.PFS2*). These files are written in ASCII format for compatibility across platforms.

5.6.4 Solve/Solution

The 'Solution' option from the Solve menu allows the user to setup an application, and start and stop the application.

5.6.4.1 Solve/Solution/Setup Environment

The ‘Setup Environment’ option allows the user to specify whether the ALLSPD-3D executable has been compiled for ‘Serial’ or ‘Parallel’ mode. It also allows the user to specify the job submission mode (Interactive or Batch), and the host selection mechanism.

In interactive mode, the user can specify the hosts to be used, or allow LSF to select the hosts to be used, subject to user specified resource requirements. Refer to the LSF manual for details on specifying resource requirements. An example resource string might be

cs && (type == RS6K)

which might mean that the user wants an RS6000 compute server. In batch mode, the user can submit a job to a specified LSF queue. Again, refer to the LSF manual for details on specifying batch queues.

If the user manually selects hosts to run the application, the GUI searches for a host list specifying existing valid hosts. The XMALLSPD_HOST_LIST environment variable defines the location of this host list. If this environment variable has not been set, the GUI follows a search path through:

.xmallspd/host_list

$HOME/.xmallspd/host_list

If these files cannot be found, the GUI creates a new file in *.xmallspd/host_list* using the current host as the initial entry.

5.6.4.2 Solve/Solution/Solve

The ‘Solve’ option allows the user to specify whether the grid pre-processor should be run, and actually submits the job to the selected environment. Note that at this time, the ALLSPD-3D executable must be compiled to match the grid size and number of species selected by the GUI. Eventually the GUI may query the code to verify that the ALLSPD-3D code matches the GUI inputs, however this has not been implemented at this time.

The Solve option starts a *'run-allspd'* script which is located in the $XMALLSPD_ROOT/bin directory. The *run-allspd* script uses *'allspd3d'* as the default name for the ALLSPD-3D executable. This can be modified by setting the XMALLSPD_EXEC environment variable. *'pregrid'* is the default name for the grid pre-processor. Again, this can be modified by setting the XMALLSPD_PREGRID environment variable.

The GUI creates three 'temporary' input files when the 'SOLVE' button is pushed. These files reflect the current values entered via the GUI. These files are named *JOB.allspd3d_input.<pid>*, *JOB.allspd3d_prop.<pid>*, and *JOB.allspd3d_link.<pid>*, where <pid> is the UNIX process id of the *run-allspd* script. These files are written to the current working directory and are used as input to the ALLSPD-3D code.

5.6.4.3 Solve/Solution/Stop

At the time of writing this manual, the Stop option has not been fully implemented. Currently the user may stop a serial or parallel job which is started on the local host (i.e., the host which is running the GUI), and a parallel job which is started on a remote host. Jobs submitted to a batch scheduler should be stopped by using the batch scheduler stop mechanism. Serial jobs which are started on remote hosts must be killed manually using the UNIX 'kill' command.

5.6.5 Solve/Help

The help files are included in the GUI. If a help file does not currently exist, pressing that 'Help' button will have no affect.

5.7 View

The View section of the main menu gives the user access to post-processing packages. Currently the GUI allows starting FAST, and the CFD-VIEW package developed by CFD Research.

The user may set the XMALLSPD_CFDVIEW_EXEC environment variable to point to the location of the CFD-VIEW executable. If this environment variable is not set it is assumed that the default name 'CFD-VIEW' is used, and that the executable is in the user's path. It is assumed that licenses to access CFD-VIEW are in place, and that CFD-VIEW will run on the same host as the GUI.

The user may set the XMALLSPD_FAST_EXEC environment variable to point to the location of the FAST executable. If this environment variable is not set it is assumed that the default name 'fast' is used, and that the executable is in the user's path. It is assumed that licenses to access FAST are in place, and that FAST will run on the same host as the GUI.

5.8 Help

Most of the help files are included in the GUI, however some have not. If a help file does not currently exist, pressing that 'Help' button will have no affect.

5.9 Known Bugs

Several locations in the GUI require a user to hit an "Add" button before entering data. If the user enters data without hitting Add, the data appears to be entered, but it is actually lost. This must be modified to prevent a user from entering data if a specific item has not been added.

5.10 Future Plans

The following items may be modified in the future:

The assumptions regarding the GUI host and all compute nodes sharing a common file system should be relaxed to allow an arbitrary file system.

Currently an entry cannot be deleted from the species database or the default chemistry packages. More flexibility to manipulate these databases should be added.

VI : Code Structure and Hardware Implementations

6.1 General Structure

The ALLSPD-3D code is not currently considered a "black box" production code, however, it is not a pure "research" code either. During the course of this development, we have tried very hard to run test cases by modifying only the user interface files, not the source code. The current version of the code can run in either serial or parallel mode. In other words, it can run on a Cray-YMP, a single workstation (SGI, IBM, HP,...), or a cluster of workstations. Also, the Graphical User Interface (GUI) allows users to build input data files (grid, chemistry, flow conditions) and initiate interactive post-processing. This section will discuss these features. The ALLSPD-3D code consists of a main program and a set of subroutines. The general structure illustrated in Figure 10 shows a flowchart encompassing the entire calculation. Beside each box in the flow diagram appears the name(s) of the primary subroutine(s) responsible for the associated task. Short comments in each subroutine source listing also describe its function.

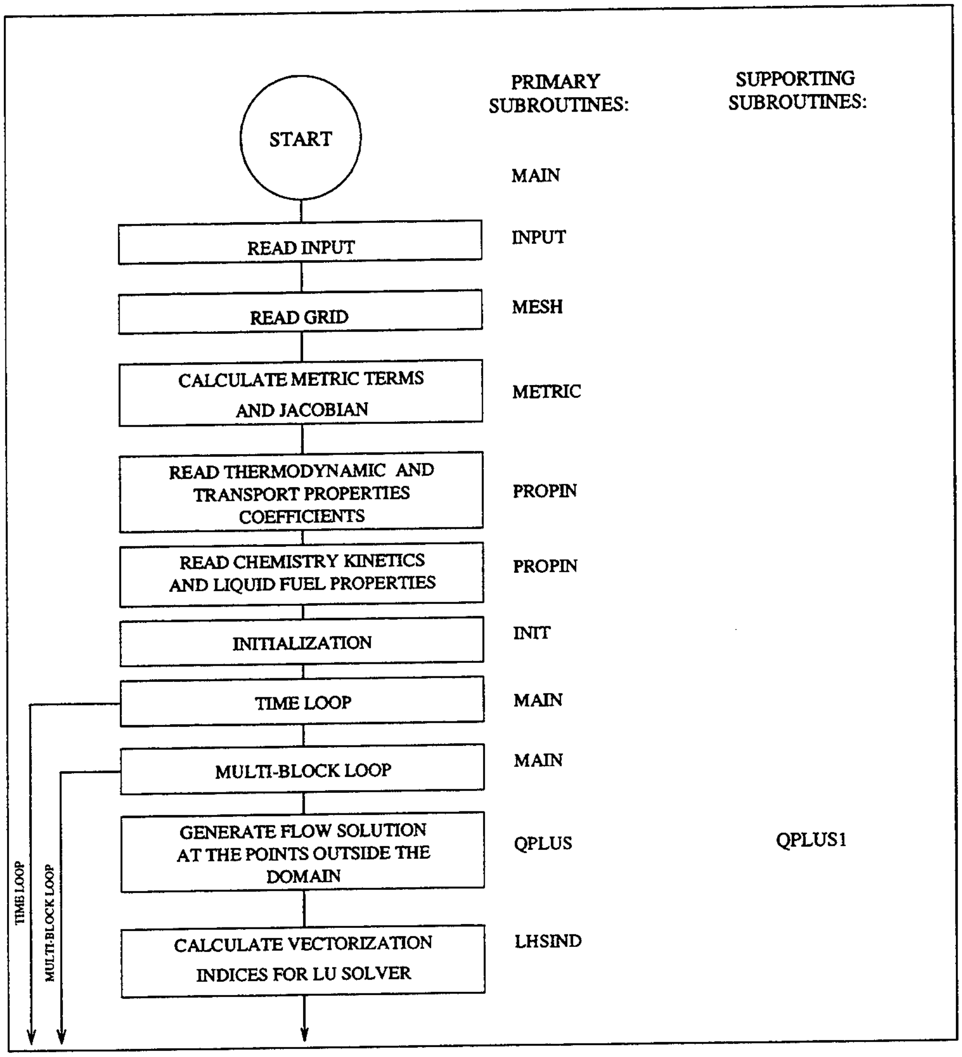

PRIMARY SUBROUTINES:
SUPPORTING SUBROUTINES:
START
MAIN
READ INPUT
INPUT
READ GRID
MESH
CALCULATE METRIC TERMS AND JACOBIAN
METRIC
READ THERMODYNAMIC AND TRANSPORT PROPERTIES COEFFICIENTS
PROPIN
READ CHEMISTRY KINETICS AND LIQUID FUEL PROPERTIES
PROPIN
INITIALIZATION
INIT
TIME LOOP
MAIN
MULTI-BLOCK LOOP
MAIN
GENERATE FLOW SOLUTION AT THE POINTS OUTSIDE THE DOMAIN
QPLUS
QPLUS1
CALCULATE VECTORIZATION INDICES FOR LU SOLVER
LHSIND
TIME LOOP
MULTI-BLOCK LOOP

FIGURE 10. **ALLSPD-3D Flow Chart**

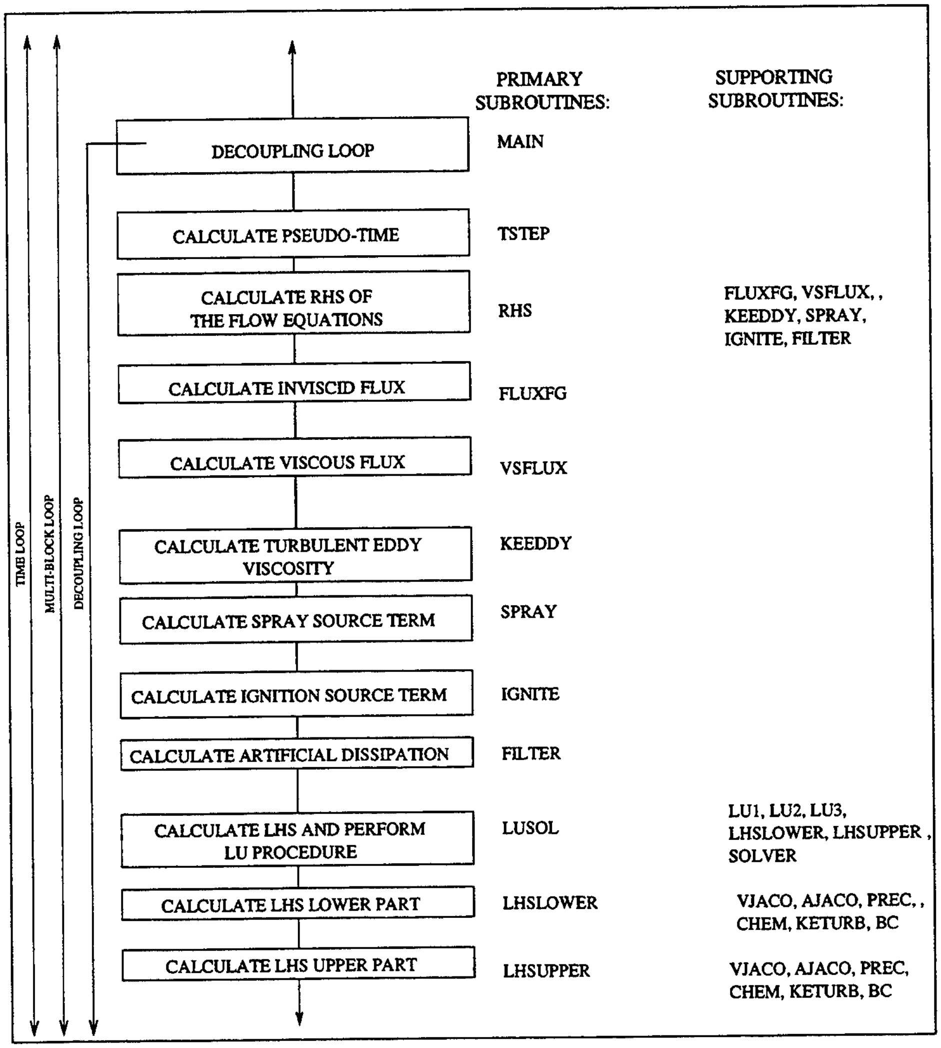

PRIMARY SUBROUTINES:
SUPPORTING SUBROUTINES:
DECOUPLING LOOP
MAIN
CALCULATE PSEUDO-TIME
TSTEP
CALCULATE RHS OF THE FLOW EQUATIONS
RHS
FLUXFG, VSFLUX, , KEEDDY, SPRAY, IGNITE, FILTER
CALCULATE INVISCID FLUX
FLUXFG
CALCULATE VISCOUS FLUX
VSFLUX
CALCULATE TURBULENT EDDY VISCOSITY
KEEDDY
CALCULATE SPRAY SOURCE TERM
SPRAY
CALCULATE IGNITION SOURCE TERM
IGNITE
CALCULATE ARTIFICIAL DISSIPATION
FILTER
CALCULATE LHS AND PERFORM LU PROCEDURE
LUSOL
LU1, LU2, LU3, LHSLOWER, LHSUPPER , SOLVER
CALCULATE LHS LOWER PART
LHSLOWER
VJACO, AJACO, PREC, , CHEM, KETURB, BC
CALCULATE LHS UPPER PART
LHSUPPER
VJACO, AJACO, PREC, CHEM, KETURB, BC
TIME LOOP
MULTI-BLOCK LOOP
DECOUPLING LOOP

FIGURE 11. **ALLSPD-3D Flow Chart (cont'd)**

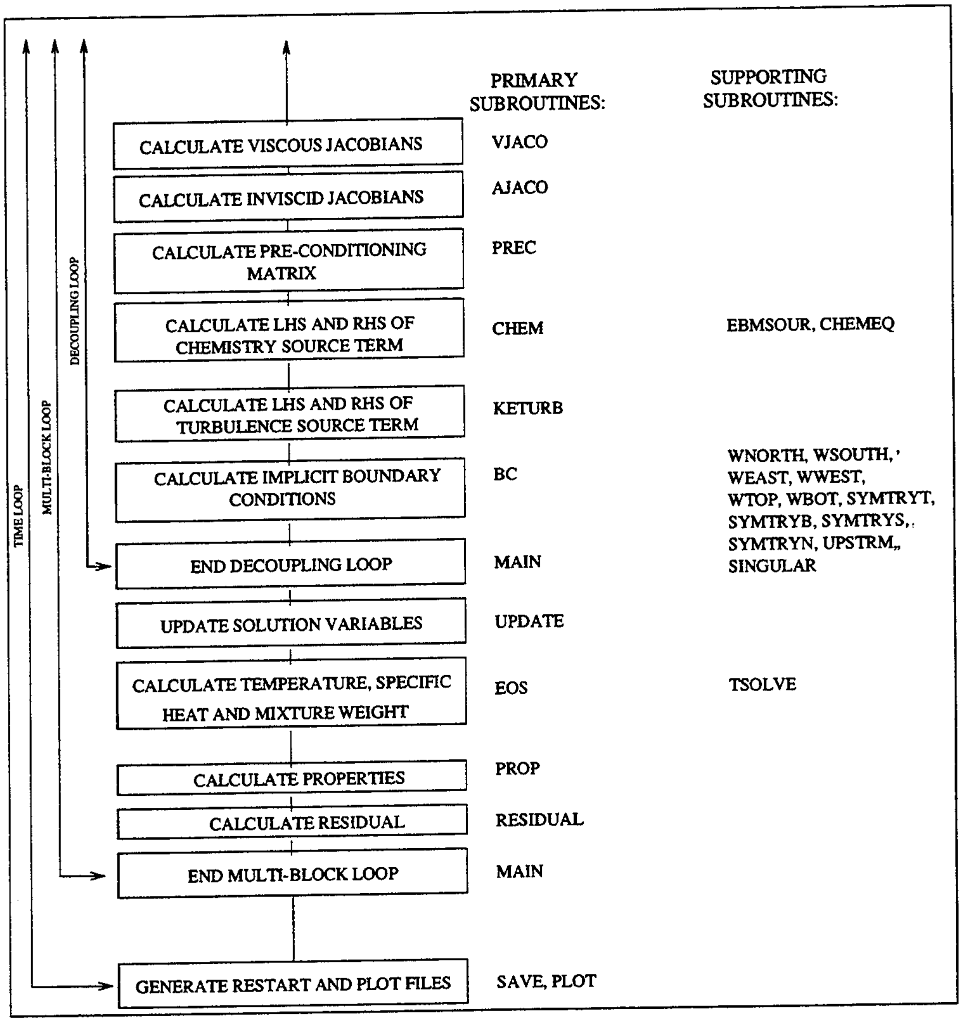

PRIMARY SUBROUTINES:
SUPPORTING SUBROUTINES:
TIME LOOP
MULTI-BLOCK LOOP
DECOUPLING LOOP
CALCULATE VISCOUS JACOBIANS
VJACO
CALCULATE INVISCID JACOBIANS
AJACO
CALCULATE PRE-CONDITIONING MATRIX
PREC
CALCULATE LHS AND RHS OF CHEMISTRY SOURCE TERM
CHEM
EBMSOUR, CHEMEQ
CALCULATE LHS AND RHS OF TURBULENCE SOURCE TERM
KETURB
CALCULATE IMPLICIT BOUNDARY CONDITIONS
BC
WNORTH, WSOUTH,' WEAST, WWEST, WTOP, WBOT, SYMTRYT, SYMTRYB, SYMTRYS,, SYMTRYN, UPSTRM,, SINGULAR
END DECOUPLING LOOP
MAIN
UPDATE SOLUTION VARIABLES
UPDATE
CALCULATE TEMPERATURE, SPECIFIC HEAT AND MIXTURE WEIGHT
EOS
TSOLVE
CALCULATE PROPERTIES
PROP
CALCULATE RESIDUAL
RESIDUAL
END MULTI-BLOCK LOOP
MAIN
GENERATE RESTART AND PLOT FILES
SAVE, PLOT

FIGURE 12. ALLSPD-3D Flow Chart (cont'd)

6.2 Working Units

The M-K-S system of units is used for all variables throughout the code. The units for several variables are listed as follows.

Length: meter (m)

Time: second (s)

Mass: kilogram (kg)

Temperature: Degrees Kelvin (K)

Energy: Joules (J)

Force: Newton (N)

Velocity: meter/second (m/s)

Pressure: N/m^2

Density: kg/m^3

Specific heat: J/(kg K)

Dynamic viscosity: kg/(m s)

Kinematic viscosity: m^2/s

The dimensional units required for chemical kinetic rate pre-exponents A_f and A_b are in the M-K-S system. Typically, the activation energy is normalized by the ideal gas constant and represented as an activation temperature in the code. The required dimensional units for these quantities are,

$$[A_f(k)] \text{ or } [A_b(k)] = \left[\frac{\text{kmols}^{1-\text{rm}_k}\,\text{m}^{3\text{rm}_k\text{-}3}}{\text{sec}}\right] \qquad (80)$$

where,

$$\mathrm{rm}_k = \sum_{i=1}^{N} \mathrm{v'}_{ik} \quad \text{or} \quad \sum_{i=1}^{N} \mathrm{v''}_{ik} \tag{81}$$

respectively, and

$$\left[\frac{E_f}{\mathrm{R_u}}\right] \text{or} \left[\frac{E_b}{\mathrm{R_u}}\right] = [K] \tag{82}$$

The ALLSPD-3D finite chemistry modules have been developed with a generalized chemistry formulation. Therefore, any finite rate and global chemistry models that can be fitted into the Arrhenius form shown in Equation (25) can be used with the code. The method of inserting various chemical reaction mechanisms is shown in the Input/Output Files section along with several examples.

6.3 Coordinate System

The governing equations in the code are transformed from the original Cartesian components to generalized coordinates. Either right-hand or left-hand rule based coordinates are acceptable for an ALLSPD-3D calculation, although the right-hand rule is the standard convention and most commercial grid generation tools will not allow the use of left-handed grid system. Figure 13 shows two typical examples of acceptable grid systems. Here, physical (x-y-z) coordinate system, the block numbers and transformed coordinates (ξ-η-ζ) are illustrated as a reference. Although there is no particular restriction on the primary flow direction in the ALLSPD-3D code, the ξ direction is preferred.

FIGURE 13. Coordinates

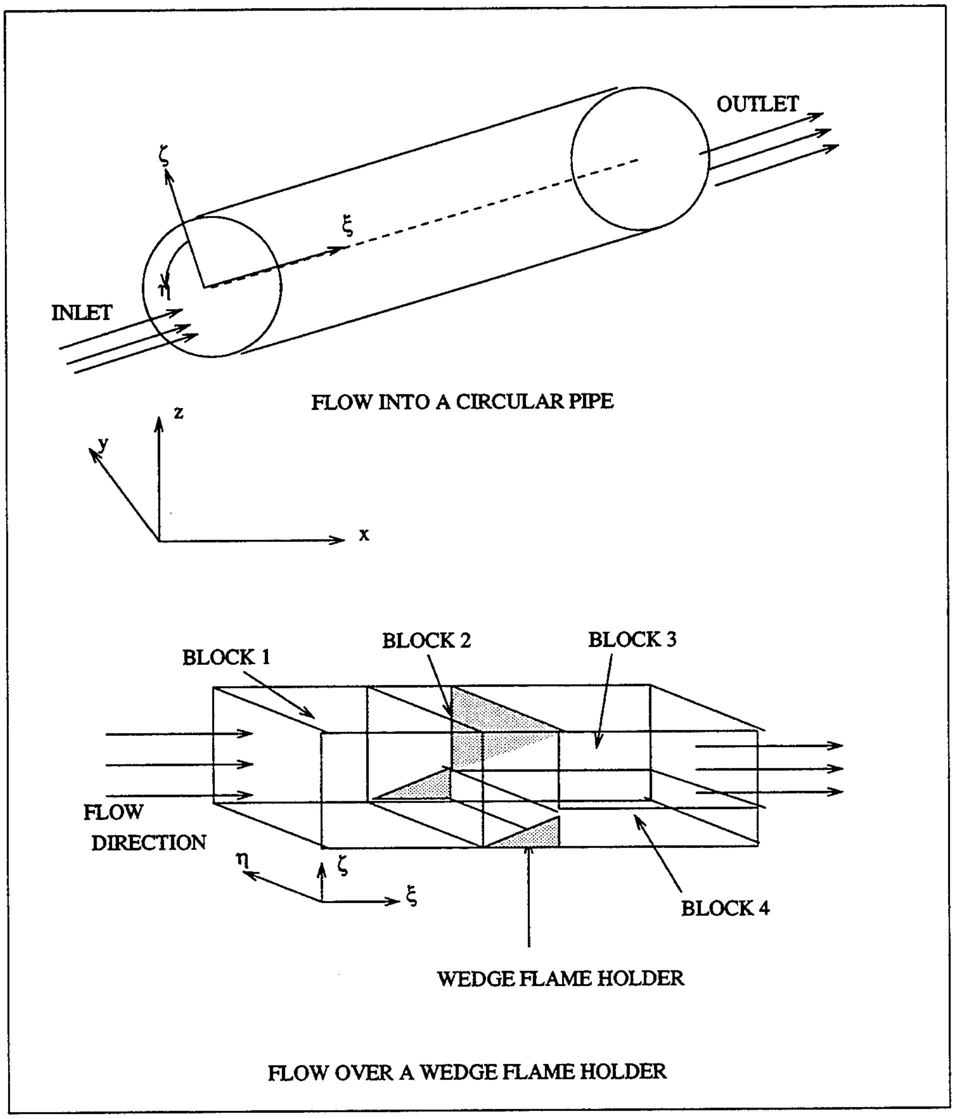
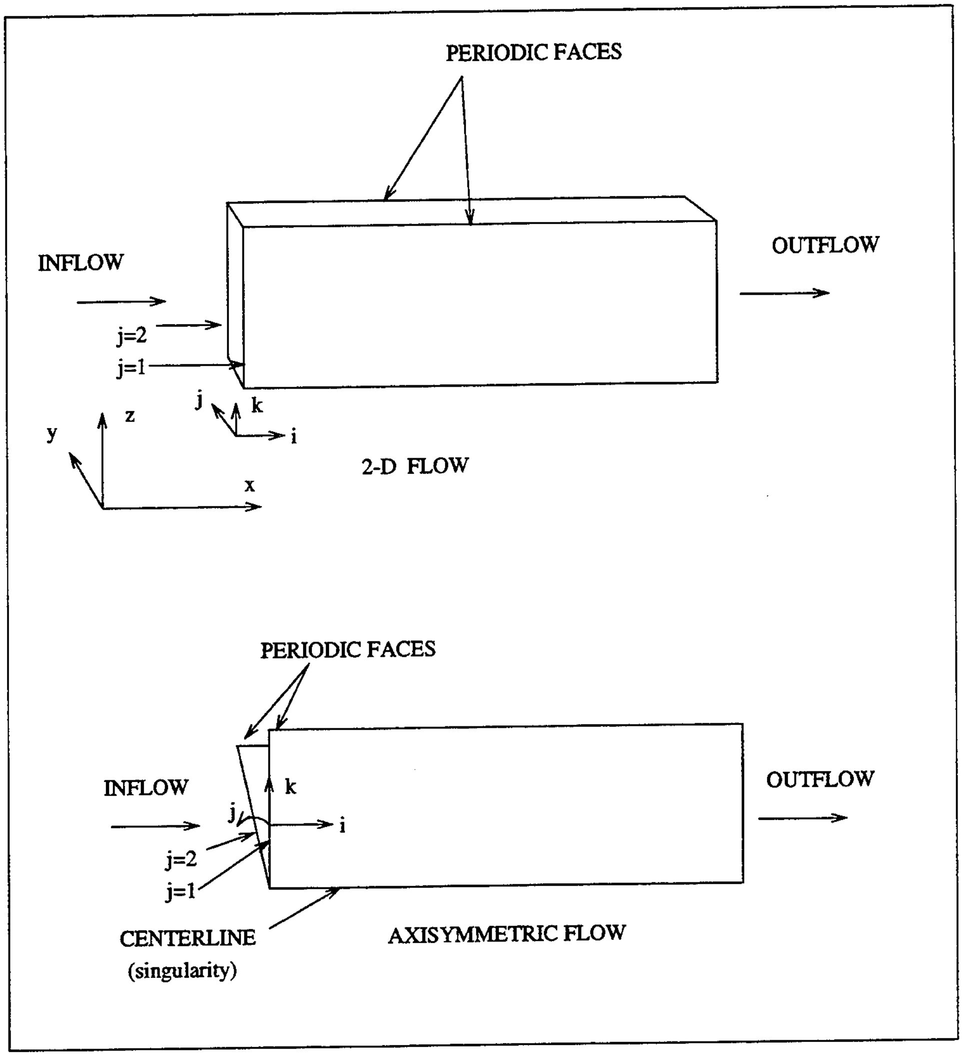


FIGURE 14. Two Dimensional Planar and Axisymmetric Configurations



6.4 ALLSPD-3D For 2-D Problems

The current ALLSPD-3D code can be used to compute two-dimensional or axisymmetric flows. The optimal method for calculating 2-D or axisymmetric flows is to use the periodic boundary conditions, which require a minimum of 2 grid points in the periodic direction. In the current version of the code, it is suggested that the periodic boundary condition be used in the *j* or η direction, if possible. (See the I/O section for details on defining periodic boundaries.) Users must generate either two parallel grid points for a 2-D planar case or 2 grid points with an angle for an axisymmetric case (see Figure 14).

6.5 ALLSPD-3D Code Compilation

The code is designed to run on different computer platforms in either serial or parallel modes. The compilation for different machines is controlled by the *Makefile* using the proper *make* commands, which are described in this section. But before doing that, users must define some parameters required to define the problem size. The parameters in the *allspd.h*, which resides in the subdirectory *CFD/src/Include*, are used to define the problem size. These parameters are described as follows.

I_MAX: The maximum number of grid points in the *i* direction among all blocks.

J_MAX: The maximum number of grid points in the *j* direction among all blocks.

K_MAX: The maximum number of grid points in the *k* direction among all blocks.

IJK_TOTAL: $\Sigma_{n=1}^{nblk}(imax_n+2)\times(jmax_n+2)\times(kmax_n+2)$ where $imax_n$, $jmax_n$, and $kmax_n$ are the grid points of the n*th* block in *i, j, k* respectively.

NUM_ZONES: Number of blocks defined in the domain.

NUM_SPECIES: Number of species.

NUM_REACT: Number of reaction equations.

MAX_OBSTACLES: Maximum number of internal blockages.

MAX_DILUTION: Maximum number of dilution holes.

MAX_TRANSPIR: Maximum number of transpiration holes.

MAX_SPARKS: Maximum number of ignition sparks.

MAX_SPRAY_PARTICLES: Maximum number of spray particles.

MAX_SPRAY_INTEGRATION: Maximum number of integration time steps for the spray particles.

The *make* command is utilized for easy compilation on a variety of computer platforms. But before compiling for a new test case on any machine, run *make veryclean* to get rid of any old files leftover from previous compilations. The *make* command targets for both serial and parallel versions follow.

TABLE 9. The *make* Commands for Supported Platforms

Platform	Serial Code	Parallel Code
Cray	*make cray*	N/A
HP-UX	*make hppa*	*make p_hppa*
IBM RS/6000	*make rs6k*	*make p_rs6k*
SGI/IRIX5	*make sgi5*	*make p_sgi5*
SUN/SunOS4.1	*make sun4*	*make p_sun4*
SUN/Solaris2	*make sun4sol2*	*make p_sun4sol2*

6.6 Program Execution (sans GUI)

After compiling the code, you can use the *make* command to install the executables *allspd3d* and *pregrid* wherever you intend to run your test case. This is done quite simply by utilizing the **DEST** definition in the *Makefile* with the *install* target. For example, if you would like to install the ALLSPD-3D executables in the *../../Tests/Swirl-combustion* directory (relative to where you built the code), you would run *make DEST=../../Tests/Swirl-combustion install* after a successful compilation.

Once the executables are installed, you need to create the various input files needed for your test case. At a minimum, the *allspd.dat*, *grid.dat*, *gridlink.dat*, and *prop.dat* files (or their equivalents named according to your preferences) must exist in the test directory. Other input files may be needed based upon the details of your particular test case. (See the **File I/O** section for details on file descriptions, formats, etc.) After you have all the needed files in place, you can run the code.

If your test case utilizes the turbulence model, you must first run *pregrid* which will pre-process the grid and generate a file called *ynbl.dat*. The *ynbl.dat* file contains normal distances (**yn**) to the nearest wall for each point in the grid. It also contains a crude estimate of the largest dimension of the grid (**xlmax**) for initialization of the turbulence quantities. The options for *pregrid* are listed below with the default file names which will be used in their absence.

TABLE 10. The *pregrid* Command Line Options

option	default file name	example
-O	*out-grid.dat*	-O*out-grid.dat*
-f	*gridlink.dat*	-f*gridlink.dat*
-g	*grid.dat*	-g*grid.dat*
-y	*ynbl.dat*	-y*ynbl.dat*

During the execution of *pregrid*, various bits of information will appear on the standard output to inform you of the pre-processor's progress as it reads and processes *grid.dat*. When *pregrid* is finished, the *out-grid.dat* file will contain a synopsis of the data found in *gridlink.dat* for your verification. This pre-processing step is needed only once unless you change the grid. If you decide to restart a test case, there is no need to re-run *pregrid*.

Now you are ready to run the flow solver. In the simplest case, all you have to do is run *allspd3d*.The various options for using alternate file names are listed in the following table. (Again, see the **File I/O** section for details on file formats and descriptions.)

TABLE 11. The ***allspd3d*** Command Line Options

option	default file name	example
-i	*allspd.dat*	**-i***allspd.dat*
-o	*out.dat*	**-o***out.dat*
-r	*res.dat*	**-r***res.dat*
-k	*reske.dat*	**-k***reske.dat*
-p	*allspd*	**-p***allspd*
-s	*save.dat*	**-s***save.dat*
-F	*face.dat*	**-F***face.dat*
-g	*grid.dat*	**-g***grid.dat*
-f	*gridlink.dat*	**-f***gridlink.dat*
-u	*uvwt.dat*	**-u***uvwt.dat*
-l	*spray.dat*	**-l***spray.dat*
-c	*prop.dat*	**-c***prop.dat*
-y	*ynbl.dat*	**-y***ynbl.dat*

As *allspd3d* starts up, it reads the various input files and reports a synopsis to *out.dat* and standard output. As the calculation progresses, numerous messages will appear on the standard output and in various output files. When the code finishes, your directory will contain output files and (if selected) restart files.

To continue the calculation, you need to edit *allspd.dat* accordingly and execute *allspd3d* again. The code will then read in the files *save.dat* and *face.dat,* in addition to all the other inputs. This will include the *ynbl.dat* file previously generated by *pregrid* if a turbulent flow is being simulated. You should see the code report a successful restart message after reading all the necessary files. After this message, the code executes as normal.

6.7 Parallel Implementation

6.7.1 Theory

The theory behind the parallel processing version of ALLSPD-3D is quite simple: the code is inherently divided in the data domain, therefore domain decomposition is used. (How's that for simplicity?) The multiple grid zone feature provides natural dividing lines in the data for decomposing the problem onto multiple processes, i.e., each grid zone is a natural candidate for parallel processing. Choosing which data to send between processes is also quite simple: essentially whatever variables are copied out of the 1-D arrays and into the 3-D arrays during *CALL CONV13* in the serial code. This data is transferred between processes using a message-passing library instead of shared memory to maximize portability.

The transferred data supplies exactly the information a process needs from its neighboring zones without requiring memory space for all of the data in other zones. Because each process needs data for only its own grid zone (including those ghost cells which actually belong to neighboring zones), each process only needs enough memory for the largest zone. (ALLSPD-3D sizes all of its 3-D data arrays based on the largest zone in the problem.) This reduced memory feature of parallel processing can be very beneficial with large problem sizes. Also, since each process only calculates data on its zone, the time needed to calculate a single iteration is reduced to (more or less) the time needed for the most numerically intensive zone. The only cost for these great benefits of parallel processing is the time it takes to transfer data between neighbors.

6.7.2 Implementation

The PVM (Parallel Virtual Machine) message-passing library was chosen because of its wide acceptance, installed user base, and portability. PVM is used in a wide variety of applications on numerous architectures and has become a de-facto standard for message-passing libraries. The PVM library has many features including spawning of processes on a virtual machine and the communication of various message types between architectures which may have inherently different data structures. These features are used in the parallel version of ALLSPD-3D. (For more information on PVM, visit *http://www.epm.ornl.gov/pvm/.*)

When the parallel version of ALLSPD-3D is executed, it automatically spawns a child process for each grid zone except the one zone that the parent runs. The parent process then becomes the controlling node in the parallel calculation and collects standard output from all the children for display in the controller's output. (This can produce some ugly output, but it is still the best way to gather all the information.) Each process then runs as a single zone calculation with needed data passed between neighbors across the zone interfaces.

The data that is transferred between grid zones (and therefore processes) includes both geometric data and flow data. The geometric data is passed between neighbors twice at startup: first to transfer X, Y, Z coordinate information about neighboring grids, then to update the various derived quantities. This is necessary since each process reads in the entire grid, but only stores the data for its own zone. Then the flow information (pressure, velocities, enthalpy, species concentrations, viscosity, etc.) is transferred between neighbors after initialization and after each iteration. To minimize the size of each message, the data is transferred in individual messages, e.g., one message for X values, another for U velocity values, etc. (All these messages are sent using PVM's default data format which allows a heterogeneous mix of machines to transparently share data without concern for data representation details on each architecture.) All this message passing fills in the data at the ghost cells for each zonal interface.

Sending data to neighbors after each iteration impacts the behavior of the code both superficially and significantly. The transfer of data obviates any need to move data in and out of the 3-D arrays. It also forces the ghost cells to contain data which is lagged a full iteration, i.e., the data one zone uses from a neighbor is old information. In the serial version of ALLSPD-3D, the zone interfaces can use data updated during the current iteration depending upon the zone ordering. (This fully-lagged boundary condition at the zone interfaces is not necessarily bad. Some test cases have actually had improved convergence in parallel processing, presumably from the dampening of errors across lagged boundaries.) This is a fundamental change in the propagation of information through the flow field and results in differing convergence histories for the parallel and serial versions of ALLSPD-3D.

By sending and receiving data between neighbors and then synchronizing with only neighboring zones, a loose synchronization of the total calculation is achieved. Basically, the process that takes the longest to calculate its zone will force all the others to wait for it because of the interdependence of the grid zones. This also means that load balancing will be best achieved by dividing the grid into more or less equally sized zones. (The only caveat here is that the spray model requires additional resources and therefore fewer grid points should be used in the zone containing spray calculation to achieve load balance.)

6.7.3 Notes About Coding Details

The ALLSPD-3D code is primarily written in Fortran, but much of the parallel coding is in C. This results in the sharing of some data between C and Fortran which is not pretty. Fortunately, the basic data transfer of geometric and flow data is handled by Fortran code (to avoid the perverse practice of accessing Fortran COMMON BLOCK data from C) and the parallel management is performed by C code.

If you decide to modify the basic sequencing of sends, receives, and synchronizations, be very careful. The checking for abort signals is critical to avoid deadlocking the calculation because one node has decided to exit while another is waiting for some message that will never arrive.

The amount of data transferred between neighbors is based on the largest possible size of a face because of limitations in Fortran. This means that messages could be greatly oversized for particular problems, especially 2-D calculations. If you wish to reduce the data passed between zones to a more reasonable size for a specific test case, you can change the definition of MAXFACE in the *CFD/src/Include/allspd.h* file. (But be very careful.) Also, the communication time for data transfer can be improved for particular test cases. For generality, the parallel version of ALLSPD-3D uses the default data transfer mode of PVM. If your virtual machine is homogeneous, you could switch to raw data mode or data in place mode. Just look for PVM_DATA_XFER_C and PVM_DATA_XFER_F in the *CFD/src/Include/allspd.h* file to change the data transfer mode.

Currently, ALLSPD-3D has no provision for spawning children to specific machines based upon CPU speed, grid zone size, etc. Therefore, the gridding will have a direct influence on the load balance of the overall calculation. Also, the PVM daemons will decide which host will calculate which zone, usually without regard to existing CPU loads. This means that you can easily wind up with processes spawned in the worst (or best) possible allocation, especially in highly heterogeneous virtual machines. Another impact from PVM is the connection used between machines. A non-shared communication medium (e.g., ATM, high-speed switches, etc.) is best. Standard ethernet is very common, but it is also the slowest, especially when all the processes need to communicate at the same time.

6.7.4 Compilation and Run-time Details

To compile and run the parallel processing version of ALLSPD-3D, you must have PVM 3.3 installed on all hosts that you intend to use. The *Makefile* uses the environment variable **PVM_ROOT** and **PVM_ARCH** to know where PVM include files and libraries are located, so make sure PVM is set up properly. Then compilation is just a matter of issuing the proper command. (I recommend that you always run *make veryclean* whenever starting a new test case or building for a different architecture.) Assuming you want to run ALLSPD-3D in parallel on a cluster of IBM RS/6000 workstations, you would run *make* *p_rs6k.* The *make* command will then compile the various modules of ALLSPD-3D including those needed for parallel processing. The pre-processing of the source code will automatically include those sections of code needed for parallel processing and leave out sections not needed. If your cluster of workstations is heterogeneous, just build with the correct parallel target (e.g., *make p_sgi* for SGI machines) and install the resulting executables in separate directories according to the needs of PVM. (And don't forget to run *make veryclean* between builds.)

After the executable(s) are built and installed in the proper locations, run *pregrid* as normal, then start PVM with a *hostfile* which is configured for your particular cluster and your problem. (This is basically a PVM issue and will not be covered here except to say that the **ep** [executable path] and **wd** [working directory] options in the *hostfile* are critical. Note that the GUI will handle these details for you.) After the pvm daemons are running on all the hosts, you start up ALLSPD-3D pretty much as normal. Be careful not to use pathnames on heterogeneous virtual machines since ALLSPD-3D will spawn processes on the virtual machine using the same command line you issue. After this, each process will read all the input files but only use the parts it needs, e.g., each process will read the entire grid but store only its zone. Whenever a process writes out an output file, it uses the name supplied on the command line (or the default name) with a grid zone extension, i.e., the process calculating zone 1 will output its turbulence residual data for its zone (and only its zone) to *reske.dat-001* instead of *reske.dat.* A slight exception to this rule is the global residual file *res.dat.* Each process gets its own residual file for the flow equations on its zone. The controlling node, however, also outputs a residual for the entire flow field to *res.dat* after gathering information from all the processes and calculating a residual spanning the entire grid. After all the processes finish the calculation, you should probably halt the PVM daemons.

When the code finishes, you will have numerous files because of each process dumping out its own results. The solution files are single zone solutions in multi-zone Plot3D format, but can be easily assembled into a single file. The restart files (*save.dat-nnn* and *face.dat-nnn*) are inherently incompatible with the serial code. These files contain only the data needed for a single zone to restart the calculation while the serial code needs all the zones in a single file. Assembling the numerous restart files of a parallel calculation into a single file for use by the serial code is probably possible but likely to be a painful experience and is not recommended.

VII : Examples

This section describes two test cases computed using the ALLSPD-3D code. These cases are designed to help the users obtain the experience needed in setting up the ALLSPD-3D input data files. The ALLSPD-3D solutions are also included so that user-obtained solutions can be compared with the other computational results and also with the experimental data. Re-running these cases will help users gain experience needed to create the ALLSPD-3D inputs and is a highly recommended step for new users. The two test cases included in this section are:

(1) 3-D non-reacting laminar and turbulent transition duct flows

(2) Axisymmetric turbulent swirl-can combustor flow with spray

7.1 3-D Non-Reacting Transition Duct Flow

The fully turbulent and three-dimensional circular to rectangular transition duct problem of Davis[41] was chosen as the first test case because of the extensive and detailed experimental data available. The grid has 52,521 cells [41x21x61] and three typical streamwise stations are shown in Figure 15. The tests conditions are shown in Table 12.

Re_D	195,000
T_{ref}	298K
U_{ref}	29 m/s

TABLE 12. Reference Conditions for the Transition Duct

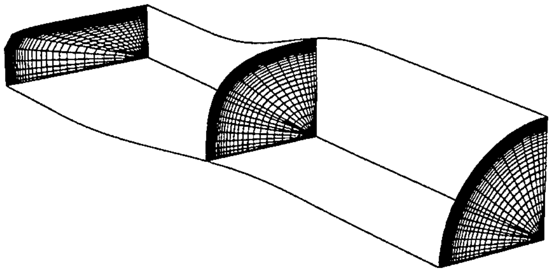


FIGURE 15. The Transition Duct Grid at Three Streamwise Stations

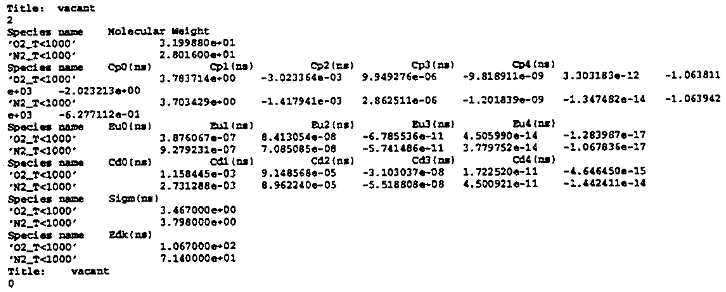
Before running the fully turbulent case, we investigated a laminar flow in the transition duct by reducing the inlet flow speed. The Reynolds number is about 195 for the laminar flow. The input data files, *gridlink.dat*, *allspd.dat*, and *prop.dat*, needed to set up this laminar flow are listed in Figure 16 to Figure 18. The velocity vectors obtained from the ALLSPD-3D solution are shown in Figure 19, and the convergence history is shown in Figure 20.

```
0
1
41 21 61
2 1 5 5 3 7 101325 0 0 0 0 0
0 0 0 0 0 0
0
0
0
```

FIGURE 16. The *gridlink.dat* Input File for the Transition Duct

```
0
2000
100
100
0 0 0
0
1
5.000000e-01 1.000000e+00
1.000000e+10 1.000000e+10 1.000000e+10
1.050000e+00 1.050000e+00 1.050000e+00
1
1.000000e-01
1e-13
1
1
0
2.995000e-02 0.000000e+00 0.000000e+00
2.970000e+02
1.013250e+05
1.021400e-01
0
0
30000
50000
0
1.600000e+03
0.000000e+00
0
0
1 2 3 4 5 11 12 13 7 9 0 0 0 0 0
2
'O2_T<1000'  2.335700e-01
'N2_T<1000'  7.664300e-01
```

FIGURE 17. The *allspd.dat* Input File for the Laminar Transition Duct



```
Title:  vacant
2
Species name    Molecular Weight
'O2_T<1000'              3.199880e+01
'N2_T<1000'              2.801600e+01
Species name    Cp0(ns)           Cp1(ns)          Cp2(ns)          Cp3(ns)          Cp4(ns)
'O2_T<1000'              3.783714e+00    -3.023364e-03    9.949276e-06    -9.818911e-09    3.303183e-12     -1.063811
e+03    -2.023213e+00
'N2_T<1000'              3.703429e+00    -1.417941e-03    2.862511e-06    -1.201839e-09    -1.347482e-14    -1.063942
e+03    -6.277112e-01
Species name    Eu0(ns)           Eu1(ns)          Eu2(ns)          Eu3(ns)          Eu4(ns)
'O2_T<1000'              3.876067e-07    8.413054e-08    -6.785536e-11    4.505990e-14    -1.283987e-17
'N2_T<1000'              9.279231e-07    7.085085e-08    -5.741486e-11    3.779752e-14    -1.067836e-17
Species name    Cd0(ns)           Cd1(ns)          Cd2(ns)          Cd3(ns)          Cd4(ns)
'O2_T<1000'              1.158445e-03    9.148568e-05    -3.103037e-08    1.722520e-11    -4.646450e-15
'N2_T<1000'              2.731288e-03    8.962240e-05    -5.518808e-08    4.500921e-11    -1.442411e-14
Species name    Sigm(ns)
'O2_T<1000'              3.467000e+00
'N2_T<1000'              3.798000e+00
Species name    Edk(ns)
'O2_T<1000'              1.067000e+02
'N2_T<1000'              7.140000e+01
Title:    vacant
0
```

FIGURE 18. The *prop.dat* Input file for the Transition Duct

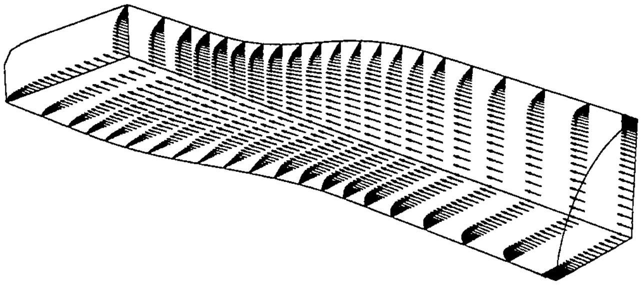


FIGURE 19. Velocity Vectors for the Laminar Transition Duct

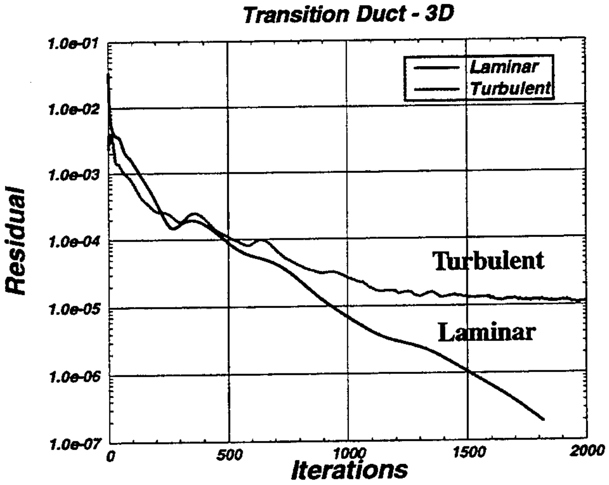


FIGURE 20. Convergence History for the Laminar and Turbulent Transition Duct

The fully turbulent transition duct flow case requires the input data file *allspd.dat* listed in Figure 17. The *gridlink.dat* and *prop.dat* are identical to those for the previous laminar case. For the turbulent flow, it is noted that **iuvwt** is equal to 1 and therefore the *uvwt.dat,* an initial condition file, has to be provided as a separate input file. Figure 22 shows the velocity vectors on the two symmetry planes of the transition duct. Comparisons of the ***u*** velocities at various streamwise stations along the two major axis with experimental data[41] are shown in Figure 23. This figure shows that reasonable agreement with experimental data can be obtained with the ALLSPD-3D code. The streamwise locations of the velocity profile comparisons shown in Figure 23 are illustrated in Figure 22. Here, additional velocity vectors on the symmetry planes are shown. Figure 20 also shows the convergence history for this turbulent flow ALLSPD-3D calculation.

```
0
2000
100
100
1 0 1
0
1
5.000000e-01 1.000000e+00
1.000000e+10 1.000000e+10 1.000000e+10
1.050000e+00 1.050000e+00 1.050000e+00
1
1.000000e-01
1e-13
1
1
1
2.995000e+01 0.000000e+00 0.000000e+00
2.970000e+02
1.013250e+05
1.021400e-01
0
0
30000
50000
0
1.600000e+03
0.000000e+00
0
0
1 2 3 4 5 11 12 13 7 9 0 0 0 0 0
2
'O2_T<1000'  2.335700e-01
'N2_T<1000'  7.664300e-01
```

FIGURE 21. The *allspd.dat* Input File for the Turbulent Transition Duct

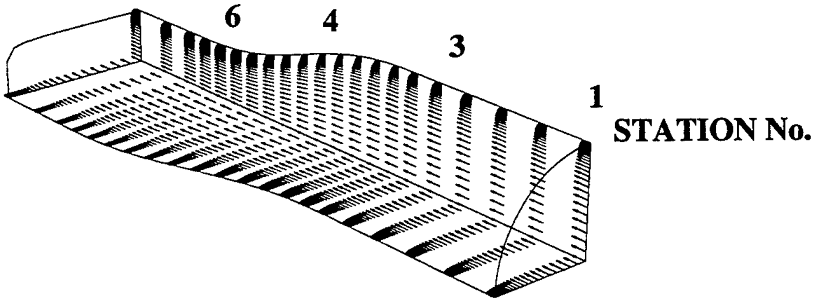
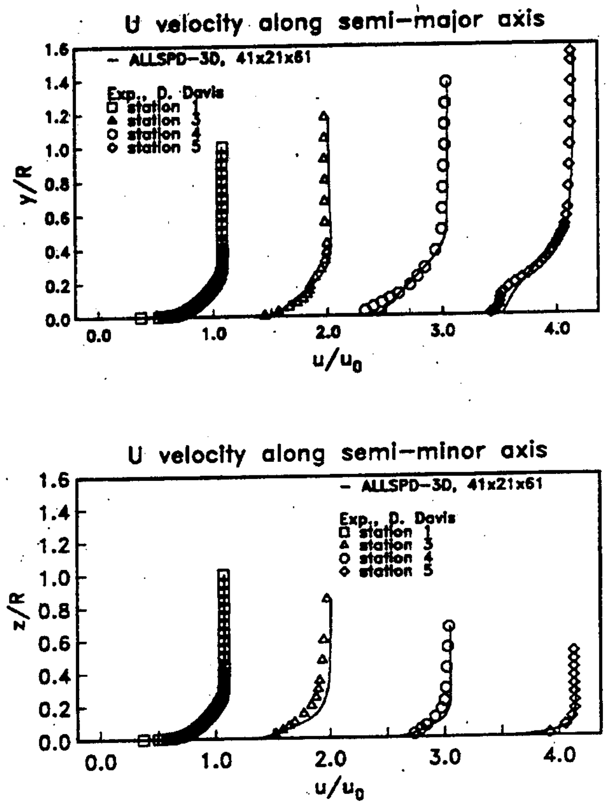


FIGURE 22. Velocity Vectors for the Turbulent Transitions Duct


U velocity along semi-major axis

U velocity along semi-minor axis

FIGURE 23. Velocity Profile Comparison

7.2 Axisymmetric Swirl-Can Combustor Flow with Spray

The second sample test case was used to validate the current turbulence model and the spray model for swirling flow. The experiment was conducted by Roback and Johnson[42] for a non-reacting turbulent flow. However, we use this case to further demonstrate the ALLSPD-3D spray model by initiating combustion in the flow field. Since this is an axisymmetric flow (see Section 6.4 of the ALLSPD-3D manual), only two grid points in the circumferential direction and the periodic boundary condition are used. Here, the combustion is initiated by injecting the liquid methanol fuel into the flow. The input data files needed are listed in Figure 24 to Figure 26. It should be noted that **iuvwt** is equal to 1 and therefore the *uvwt.dat* input file is also needed. The reference conditions are provided in Table 13.

Re_D	61,180
T_{ref}	300 K
U_{ref}	16 m/s

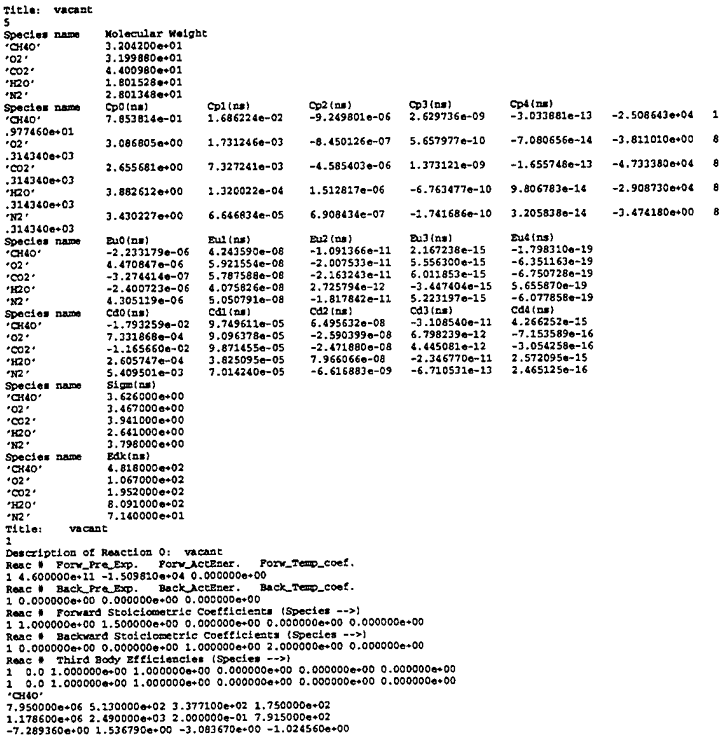
TABLE 13. Reference Conditions for the Swirl Can Combustor

```
0
6
49 2 38
15 2 38
19 2 38
49 2 24
15 2 24
19 2 24
4 1 6 6 4 7 0 0 0 0 0 0
4 4 6 6 4 7 0 0 0 0 0 0
2 4 6 6 4 7 101325 0 0 0 0 0
4 1 6 6 3 4 0 0 0 0 0 0
4 4 6 6 3 4 0 0 0 0 0 0
2 4 6 6 3 4 101325 0 0 0 0 0
2 0 0 0 4 0
3 1 0 0 5 0
0 2 0 0 6 0
5 0 0 0 0 1
6 4 0 0 0 2
0 5 0 0 0 3
0
0
0
0
0
0
0
0
0
0
0
0
0
```

FIGURE 24. The *gridlink.dat* Input File for the Swirl Can Combustor

```
0
2000
100
100
1 0 1
1
1
1.000000e-01 0.000000e+00
1.000000e+10 1.000000e+10 1.000000e+10
1.050000e+00 1.050000e+00 1.050000e+00
1
1.000000e-01
1e-13
1
1
1
1.600000e+01 0.000000e+00 0.000000e+00
3.000000e+02
1.013250e+05
6.100000e-02
0
0
300
50000
1
1 25 40 1 2 2 30
1.600000e+03
5.000000e+07
0
1
1
30 30 1 2 5 24
1 0.000000e+00 0.000000e+00
1.000000e+01 3.700000e+01
300
20
0
3.000000e+02
1.907808e-04
3.000000e+01
1 2 3 4 5 15 16 17 18 19 11 12 13 14 7
5
'CH4O'  0.000000e+00
'O2'  2.335700e-01
'CO2'  0.000000e+00
'H2O'  0.000000e+00
'N2'  7.664300e-01
```

FIGURE 25. The *allspd.dat* Input File for the Swirl Can Combustor



```
Title:  vacant
5
Species name    Molecular Weight
'CH4O'          3.204200e+01
'O2'            3.199880e+01
'CO2'           4.400980e+01
'H2O'           1.801528e+01
'N2'            2.801348e+01
Species name    Cp0(ns)         Cp1(ns)         Cp2(ns)         Cp3(ns)         Cp4(ns)
'CH4O'          7.853814e-01    1.686224e-02    -9.249801e-06   2.629736e-09    -3.033881e-13   -2.508643e+04   1
.977460e+01
'O2'            3.086805e+00    1.731246e-03    -8.450126e-07   5.657977e-10    -7.080656e-14   -3.811010e+00   8
.314340e+03
'CO2'           2.655681e+00    7.327241e-03    -4.585403e-06   1.373121e-09    -1.655748e-13   -4.733380e+04   8
.314340e+03
'H2O'           3.882612e+00    1.320022e-04    1.512817e-06    -6.763477e-10   9.806783e-14    -2.908730e+04   8
.314340e+03
'N2'            3.430227e+00    6.646834e-05    6.908434e-07    -1.741686e-10   3.205838e-14    -3.474180e+00   8
.314340e+03
Species name    Eu0(ns)         Eu1(ns)         Eu2(ns)         Eu3(ns)         Eu4(ns)
'CH4O'          -2.233179e-06   4.243590e-08    -1.091366e-11   2.167238e-15    -1.798310e-19
'O2'            4.470847e-06    5.921554e-08    -2.007533e-11   5.556300e-15    -6.351163e-19
'CO2'           -3.274414e-07   5.787588e-08    -2.163243e-11   6.011853e-15    -6.750728e-19
'H2O'           -2.400723e-06   4.075826e-08    2.725794e-12    -3.447404e-15   5.655870e-19
'N2'            4.305119e-06    5.050791e-08    -1.817842e-11   5.223197e-15    -6.077858e-19
Species name    Cd0(ns)         Cd1(ns)         Cd2(ns)         Cd3(ns)         Cd4(ns)
'CH4O'          -1.793259e-02   9.749611e-05    6.495632e-08    -3.108540e-11   4.266252e-15
'O2'            7.331868e-04    9.096378e-05    -2.590399e-08   6.798239e-12    -7.153589e-16
'CO2'           -1.165660e-02   9.871455e-05    -2.471880e-08   4.445081e-12    -3.054258e-16
'H2O'           2.605747e-04    3.825095e-05    7.966066e-08    -2.346770e-11   2.572095e-15
'N2'            5.409501e-03    7.014240e-05    -6.616883e-09   -6.710531e-13   2.465125e-16
Species name    Sigm(ns)
'CH4O'          3.626000e+00
'O2'            3.467000e+00
'CO2'           3.941000e+00
'H2O'           2.641000e+00
'N2'            3.798000e+00
Species name    Edk(ns)
'CH4O'          4.818000e+02
'O2'            1.067000e+02
'CO2'           1.952000e+02
'H2O'           8.091000e+02
'N2'            7.140000e+01
Title:    vacant
1
Description of Reaction 0:  vacant
Reac #  Forw_Pre_Exp.   Forw_ActEner.   Forw_Temp_coef.
1 4.600000e+11 -1.509810e+04 0.000000e+00
Reac #  Back_Pre_Exp.   Back_ActEner.   Back_Temp_coef.
1 0.000000e+00 0.000000e+00 0.000000e+00
Reac #  Forward Stoiciometric Coefficients (Species -->)
1 1.000000e+00 1.500000e+00 0.000000e+00 0.000000e+00 0.000000e+00
Reac #  Backward Stoiciometric Coefficients (Species -->)
1 0.000000e+00 0.000000e+00 1.000000e+00 2.000000e+00 0.000000e+00
Reac #  Third Body Efficiencies (Species -->)
1  0.0 1.000000e+00 1.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
1  0.0 1.000000e+00 1.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
'CH4O'
7.950000e+06 5.130000e+02 3.377100e+02 1.750000e+02
1.178600e+06 2.490000e+03 2.000000e-01 7.915000e+02
-7.289360e+00 1.536790e+00 -3.083670e+00 -1.024560e+00
```

FIGURE 26. **The *prop.dat* Input File for the Swirl Can Combustor**

The grid used by the ALLSPD-3D code on the swirl can combustor is shown in Figure 28. In this figure, every other point is plotted with each zone in separate colors for clarity. The grid is comprised of six blocks as detailed in Table 14. The velocity vectors obtained from the ALLSPD-3D solution at 2000 iterations is shown in Figure 29. This solution was obtained using the ALLSPD-3D code in the parallel mode where the calculation was performed on 6 workstation nodes. The velocity vectors are colored according to flow field temperature to show that the fuel has ignited. This ignition produces a flowfield where the temperature ranges from 300 K to 2100 K. In Figure 29, the color blue is used to indicate low temperature and the color red is used to indicate high temperature. The convergence history for this calculation is shown in Figure 27.

Block	Block Dimensions	Number of Points
1	49 x 2 x 38	3724
2	15 x 2 x 38	1140
3	19 x 2 x 38	1444
4	49 x 2 x 24	2352
5	15 x 2 x 24	720
6	19 x 2 x 24	912
Total		10292

TABLE 14. Swirl Can Combustor Grid Block Sizes

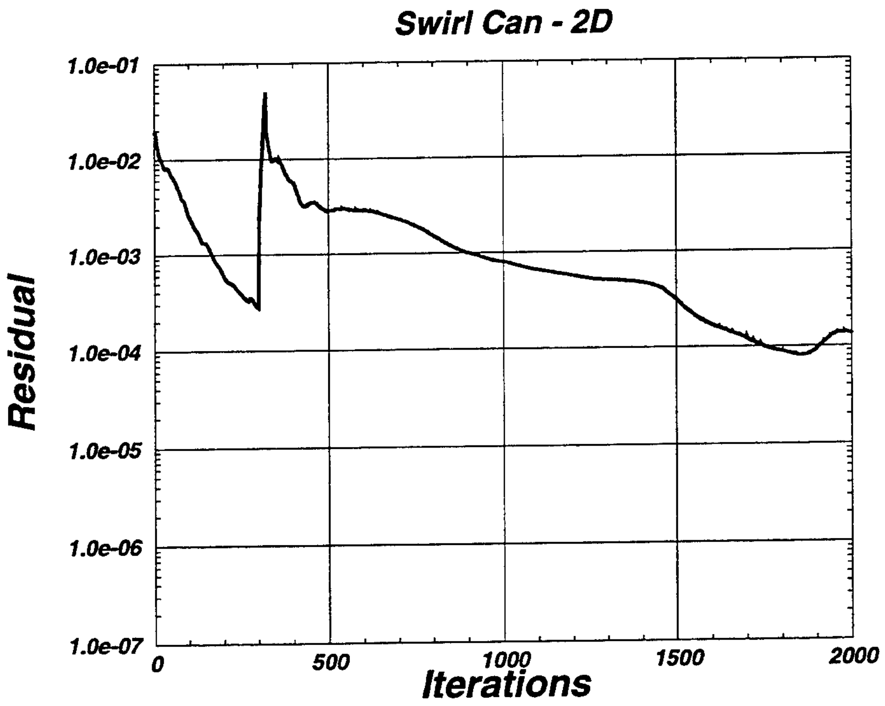


FIGURE 27. Convergence History for the Swirl Can Combustor

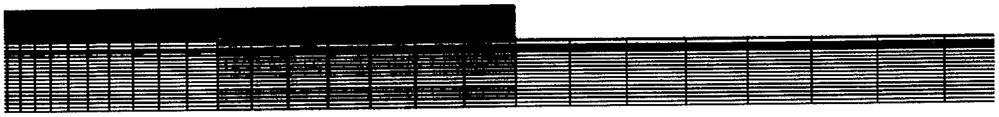


FIGURE 28. Grid for the Swirl Can Combustor

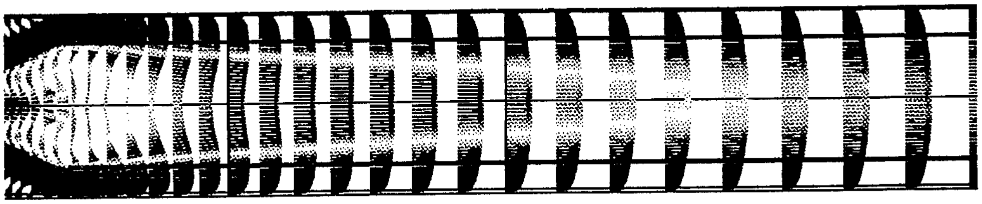


FIGURE 29. Temperature Colored Velocity Vectors

Acknowledgments

The authors would like to express deep gratitude to Dr. Robert M. Stubbs, Chief of the Computational Methods for Space Branch, and Dr. Edward J. Mularz, former Chief of the Aerothermochemistry Branch, at NASA Lewis Research Center for their coordination and guidance on the ALLSPD-3D code development. Many thanks are also due to Drs. Peter Sockol, John Deur, and Jeffrey Moder for numerous technical discussions to improve the efficiency of the code. Also, we would like to thank Bonnie McBride for helping set up the thermophysical and transport properties of the species database. Finally, special thanks to many of our colleagues at NASA Lewis, including Drs. Shaye Yungster, Suri Raju, Sheng-Tao Yu, Jian-Shun Shuen, and Nan-Suey Liu amongst others, for assistance in the early development of the code.

References

1 Bruce, T. W., Mongia, H. C. and Reynolds, R. S., "Combustor Design Criteria Validation, Vol. I- Element Tests and Model Validation," *USARTL-TR-78-551,* 1979*(also available NITS AD-A067657).*

2 Briley, W. R., McDonald, H. and Shamroth, S. J. "A Low Mach Number Euler Formulation and Application to Time-Marching LBI Scheme," *AIAA J.,* **21**, 1467,1983.

3 Turkel, E.,"Preconditioned Methods for Solving the Incompressible and Low Speed Compressible Equations," *AIAA J.,* **23**, 1518, 1985.

4 Choi, Y. H. and Merkle, C. L. "The Application of Preconditioning in Viscous Flows," *J. of Comp. Phys.,* **105**, 207,1993.

5 Chen, K.-H. and Pletcher, R. H., "A Primitive Variable, Strongly Implicit Calculation Procedure for Viscous Flows at All Speeds," *AIAA J.* **29**, No. 8, Aug., 1241, 1991.

6 Shuen, J.-S., Chen K.-H. and Choi Y., "A Coupled Implicit Method for Chemical Non-equilibrium Flows at All Speeds," *J. of Comp. Phys.*, **106**, No. 2, 306,1993.

7 Chen, K.H. and Shuen, J.-S., "Three-Dimensional Coupled Implicit Methods for Spray Combustion Flows at All Speeds," AIAA 94-3047, 1994.

8 Frenklach, M. Wang, H. and Rabinowitz, M. "Optimization and Analysis of Large Chemical Kinetic Mechanisms Using the Solution Mapping Method-Combustion of Methane," *Prog. Energy Combust. Sci.*, **18**, 47 ,1992.

9 Wang, H. and Frenklach, M., "Detailed Reduction of Reaction Mechanisms for Flame Modeling," *Combustion and Flame*, **97**, 365 ,1991.

10 Rabinowitz, M. Personal communications, NASA Lewis Research Center, Cleveland, Ohio, 1994.

11 Chen, K-H and Shuen, J.-S., "A Comprehensive Study of Numerical Algorithms for Three-Dimensional, Turbulent, Nonequilibrium Viscous Flows with Detailed Chemistry," AIAA 95-0800, 995.

12 Shih, T. H. and Lumley, J., "Kolmogorov Behavior of Near-Wall Turbulence and Its Application in Turbulence Modeling," *Comp. Fluid Dyn.*, **1**, 43,1993.

13 Magnussen, B. F. and Hjertager, B. H., "On Mathematical Modeling of Turbulent Combustion with Special Emphasis on Soot Formulation and Combustion," *16th Symposium ,Int'l on Combustion/The Combustion Institute*, 719,1976.

14 Faeth, G. M. "Evaporation and Combustion of Sprays," *Prog. in Energy and Comb. Sci.*, **9**, 1 ,1983.

15 Tong, A. Y. and Sirignano, W. A. "Analytical Solution For Diffusion and Circulation in a Vaporizing Droplet," *19th Symposium ,Int'l on Combustion/ The Combustion Institute*, 1007 ,1982.

16 McBride B., Gordon, S. and Reno, M., "Coefficients for Calculating Thermodynamics and Transport Properties of Individual Species," NASA TM 4513, 1993.

17 Reid, R. C. Prausniz, J. M. and Poling, B. E. The Properties of Gases and Liquids, fourth Ed., McGraw-Hill Publishing Co., N.Y., 1988.

18 Anderson, D. A. Tannehill, J. C. and Pletcher, R. H. *Computational Fluid Mechanics and Heat Transfer.* Hemisphere Publishing Corp., New York,1984.

19 Yoon, S. and Jameson, A, "An LU-SSOR Scheme for the Euler and Navier-Stokes Equations,", AIAA 87-600, Jan., 1987.

20 Chen, K-H and Shuen, J.-S., "A Coupled Multi-Block Solution Procedure for Spray Combustion in Complex Geometries," AIAA 93-0108, 1993.

21 Shuen, J.-S., Solomoa, A. S. P. and Faeth, G. M., "Drop-Turbulence Interactions in a Diffusion Flame," *AIAA J.*, **24**, 101,1986.

22 Shuen, J.-S., Chen, L. D. and Faeth, G. M., "Evaluation of a Stochastic Model of Particle Dispersion in a Turbulent Round Jet," *AIChE J.*, **29**, 167,1983.

23 Raju, M. S. and Sirignano, W. A. "Multicomponent Spray Computations in a Modified Centerbody Combustor," *J. of Propulsion*, **6**, 97,1990.

24 Launder B. E. and Sharma, "Application of Energy-Dissipation Model of Turbulence to the Calculation of Flow near a Spinning Disk," *Letters in Heat and Mass Transfer*, **1**, pp131-138 ,1974.

25 Chien K.Y., "Prediction of Channel and Boundary-Layer Flows with a Low-Reynolds Number Turbulence Model," *AIAA J.*, **20**, No. 1, pp. 33-38,1982.

26 Shih T. H. and Lumley J., "Kolmogorov Behavior of Near-Wall Turbulence and Its Applications in Turbulence Modelling," *NASA TM 105663*, 1992.

27 Shih T. H., Liou, W. W., Shabbir A., Yang, Z.,and Zhu J., "A New k-ε Model for High Reynolds Number Turbulent Flows-Model Development and Validation," *J. of Computers and Fluids*, Vol. 24, No.3, 1995.

28 Lee J., "Analysis of Supersonic Flows with Low-Reynolds Number Compressible Two-Equation Turbulence Models using LU Finite Volume Numerical Techniques," *AIAA 94-0193*, Jan 1994.

29 Gordon, S. and McBride, B. J., "Computer Program for the Calculation of Complex Equilibrium Composition, Rocket Performance, Incident and Reflected Shocks and Chapman-Jouguet Detonation," *NASA SP-273.*

30 Kanury, M., *Introduction to Combustion Phenomena*, Gordon and Breach,1977.

31 Valachovic T., "Numerical Prediction of Idle Power Emissions From Gas Turbine Combustors," AMSE 93-010.

32 Wilke, C.R., "A Viscosity Equation for Gas Mixture," Chem. Physics, Vol. 18, no. 4, Apr., pp. 517-519, 1950.

33 White, F, M., *Viscous Flow*, McGraw-Hill Inc., 1974.

34 Kuo K., *Principles of Combustion*, Wiley and Sons, 1976

35 Lee, J., "An Application of a Two-Equation Model of Turbulence to Three Dimensional Chemically Reacting Flows," AIAA P 95-0734, Jan., 1995.

36 Spalding B.D., “Development of the Eddy-Break-Up Model for Turbulent Combustion,” Thirteenth Symposium on Combustion, The Combustion Institute, pp. 649-657,1971.

37 Westbrook C. K. and Dryer F. L., “Simplified Reaction Mechanism for the Oxidation of Hydrocarbon Fuels in Flames,” Combustion Science and Technology, **27**, pp. 31-43, 1981.

38 Svehla R. A., “Estimated Viscosities and Thermal Conductivities of Gases at High Temperature,” *NASA TR R-132*, 1962

39 *User's Manual for GRIDGEN v10.0*, 1995

40 Walatka, P.P. and Buning P., PLOT3D User's Manual, Version 3.5, 1988, NASA Ames Research Center.

41 Davis, D., ``Experimental Investigation of Turbulent Flow Through a Circular-to-Rectangular Transition Duct,'' *NASA TM-105210*, September, 1991.

42 Roback, R. and Johnson, B. V., ``Mass and momentum turbulent transport experiments with confined swirling coaxial jets,'' *NASA CR-168252*.

Appendix A

Jacobian Matrices

The number of species equations is assumed to be five (*N*=5) in the Jacobian matrices given below

$$A^{(1)} = \begin{bmatrix} \frac{U}{RT} & \rho\alpha_1 & \rho\alpha_2 & \rho\alpha_3 & -\frac{\rho U}{C_p T} \\ \alpha_1 + \frac{uU}{RT} & \rho(U + \alpha_1 u) & \rho u\alpha_2 & \rho u\alpha_3 & -\frac{\rho uU}{C_p T} \\ \alpha_2 + \frac{vU}{RT} & \rho v\alpha_1 & \rho(U + \alpha_2 v) & \rho v\alpha_3 & -\frac{\rho vU}{C_p T} \\ \alpha_3 + \frac{wU}{RT} & \rho w\alpha_1 & \rho w\alpha_2 & \rho(U + \alpha_3 w) & -\frac{\rho wU}{C_p T} \\ \frac{UH}{RT} & \rho(uU + \alpha_1 H) & \rho(vU + \alpha_2 H) & \rho(wU + \alpha_3 H) & \rho U\phi \end{bmatrix} \tag{A 1}$$

$$A^{(2)} = \begin{bmatrix} \rho U & 0 \\ 0 & \rho U \end{bmatrix} \tag{A 2}$$

$$A^{(3)} = \begin{bmatrix} \rho U\Theta_1 & \rho UY_1\Omega_2 & \rho UY_1\Omega_3 & \rho UY_1\Omega_4 \\ \rho UY_2\Omega_1 & \rho U\Theta_2 & \rho UY_2\Omega_3 & \rho UY_2\Omega_4 \\ \rho UY_3\Omega_1 & \rho UY_3\Omega_2 & \rho U\Theta_3 & \rho UY_2\Omega_4 \\ \rho UY_4\Omega_1 & \rho UY_4\Omega_2 & \rho UY_4\Omega_3 & \rho U\Theta_4 \end{bmatrix} \tag{A 3}$$

$$U = \alpha_1 u + \alpha_2 v + \alpha_3 w \tag{A 4}$$

$$\alpha_1 = \xi_x \qquad \alpha_2 = \xi_y \qquad \alpha_3 = \xi_z \tag{A 5}$$

$$\Omega_i = -W\left(\frac{1}{W_i} - \frac{1}{W_N}\right) + \frac{(h_i - h_N)}{C_p T} \tag{A 6}$$

$$\Theta_i = 1 + Y_i \Omega_i \qquad \phi = 1 - \frac{H}{C_p T} \tag{A 7}$$

here W is the molecular weight of the gas mixture and C_p is the mixture specific heat C_{p_m}. The Jacobian matrix $B^{(m)}$ is obtained by letting $\alpha_1 = \eta_x$, $\alpha_2 = \eta_y$, $\alpha_3 = \eta_z$ and Jacobian matrix $C^{(m)}$ is obtained by letting $\alpha_1 = \zeta_x$, $\alpha_2 = \zeta_y$, $\alpha_3 = \zeta_z$.

The viscous term Jacobian has the following form

$$R^{(1)}_{\xi\xi} = \frac{1}{J}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & \mu_e\left(\frac{1}{3}\alpha_1^2 - \Lambda\right) & \mu_e\alpha_{12} & \mu_e\alpha_{13} & 0 \\ 0 & \mu_e\alpha_{12} & \mu_e\left(\frac{1}{3}\alpha_2^2 - \Lambda\right) & \mu_e\alpha_{23} & 0 \\ 0 & \mu_e\alpha_{13} & \mu_e\alpha_{23} & \mu_e\left(\frac{1}{3}\alpha_3^2 - \Lambda\right) & 0 \\ 0 & \pi_u & \pi_v & \pi_w & \frac{k_e}{C_p}\Lambda \end{bmatrix} \tag{A 8}$$

$$R^{(2)}_{\xi\xi} = \frac{1}{J}\begin{bmatrix} \left(\mu_l + \frac{\mu_t}{\sigma_k}\right)\Lambda & 0 \\ 0 & \left(\mu_l + \frac{\mu_t}{\sigma_\varepsilon}\right)\Lambda \end{bmatrix} \tag{A 9}$$

$$R_{\xi\xi}^{(3)} = \frac{1}{J}\begin{bmatrix} \rho D_{1m}\Lambda & 0 & 0 & 0 \\ 0 & \rho D_{2m}\Lambda & 0 & 0 \\ 0 & 0 & \rho D_{3m}\Lambda & 0 \\ 0 & 0 & 0 & \rho D_{4m}\Lambda \end{bmatrix} \qquad \text{(A 10)}$$

$$\begin{aligned} \alpha_{12} &= \frac{1}{3}\alpha_1\alpha_2 & \alpha_{13} &= \frac{1}{3}\alpha_1\alpha_3 \\ \alpha_{23} &= \frac{1}{3}\alpha_2\alpha_3 & \Lambda &= \alpha_1^{\ 2} + \alpha_2^{\ 2} + \alpha_3^{\ 2} \end{aligned} \qquad \text{(A 11)}$$

$$\begin{aligned} \pi_u &= \mu_e\left(u\left(\frac{1}{3}\alpha_1^2 - \Lambda\right) + v\alpha_{12} + w\alpha_{13}\right) \\ \pi_v &= \mu_e\left(v\left(\frac{1}{3}\alpha_1^2 - \Lambda\right) + u\alpha_{12} + w\alpha_{23}\right) \\ \pi_w &= \mu_e\left(w\left(\frac{1}{3}\alpha_3^2 - \Lambda\right) + v\alpha_{13} + w\alpha_{23}\right) \end{aligned} \qquad \text{(A 12)}$$

$$T^{(3)} = \begin{bmatrix} \frac{\partial\omega_1}{\partial Y_1} & \frac{\partial\omega_1}{\partial Y_2} & \frac{\partial\omega_1}{\partial Y_3} & \frac{\partial\omega_1}{\partial Y_4} \\ \frac{\partial\omega_2}{\partial Y_1} & \frac{\partial\omega_2}{\partial Y_2} & \frac{\partial\omega_2}{\partial Y_3} & \frac{\partial\omega_2}{\partial Y_4} \\ \frac{\partial\omega_3}{\partial Y_1} & \frac{\partial\omega_3}{\partial Y_2} & \frac{\partial\omega_3}{\partial Y_3} & \frac{\partial\omega_2}{\partial Y_4} \\ \frac{\partial\omega_4}{\partial Y_1} & \frac{\partial\omega_4}{\partial Y_2} & \frac{\partial\omega_4}{\partial Y_3} & \frac{\partial\omega_4}{\partial Y_4} \end{bmatrix} \qquad T^{(2)} = \begin{bmatrix} \frac{\partial S_k}{\partial k} & \frac{\partial S_k}{\partial \varepsilon} \\ \frac{\partial S_\varepsilon}{\partial k} & \frac{\partial S_\varepsilon}{\partial \varepsilon} \end{bmatrix} \qquad \text{(A 13)}$$

In order to reduce the stiffness of the chemical species source term, the source terms are treated implicitly. The chemical species source, Equation(24), is a function of temperature and species concentration and therefore a chain rule is required to evaluate the source term in general. The linearization of the left hand side leads to the following expressions:

$$\frac{\partial\omega_i}{\partial Y_j} = -\rho\left[MW\left(\frac{1}{MW_j}-\frac{1}{MW_N}\right)-\frac{(h_j-h_N)}{C_pT}\right]\frac{\partial\omega_i}{\partial\rho}$$

$$+MW_i\sum_{k=1}^{N_r}\left(\mathrm{CR}_{ik}\left(\left(K_f\frac{\nu'_{jk}}{Y_j}\right)\prod_{l=1}^{N}C_l^{\nu'_{lk}}-\left(K_b\frac{\nu''_{jk}}{Y_j}\right)\prod_{l=1}^{N}C_l^{\nu''_{lk}}\right)\right)-\frac{(h_j-h_N)}{C_{p_m}}\left(\frac{\partial\omega_i}{\partial T}\right)$$

$$\frac{\partial\omega_i}{\partial\rho} = MW_i\sum_{k=1}^{N_r}\left(\mathrm{CR}_{ik}\left(\frac{K_f}{\rho}\left(\sum_{l=1}^{N}\nu'_{il}\right)\prod_{l=1}^{N}C_l^{\nu'_{lk}}-\frac{K_b}{\rho}\left(\sum_{l=1}^{N}\nu''_{il}\right)\prod_{l=1}^{N}C_l^{\nu''_{lk}}\right)\right)$$

$$\frac{\partial\omega_i}{\partial T} = MW_i\sum_{k=1}^{N_r}\left(\mathrm{CR}_{ik}\left(\frac{K_f}{T}\left(\alpha_f(k)-\frac{E_{f_i}}{R_uT}\right)\prod_{l=1}^{N}C_l^{\nu'_{lk}}-\frac{K_b}{T}\left(\alpha_b(k)-\frac{E_{b_i}}{R_uT}\right)\prod_{l=1}^{N}C_l^{\nu''_{lk}}\right)\right)$$

(A 14)

where, $E_{f\!b}$ and $K_{f\!b}$ are function of the k^{th} index.

The turbulence source term shown in Equation(35) can be very stiff numerically. Therefore, the ALLSPD-3D code solves the equation in a point implicit manner. The linearization of the turbulent source term required the use of the following source Jacobian term. This formulation has been optimized for convergence. This form has been tested for a number of flow situations and model formulations

$$T_{ij} = \begin{bmatrix}\left(\frac{\hat{P}_k}{\rho k}-\frac{\varepsilon}{k}\right) & 0 \\ 0 & C_{\varepsilon 1}\frac{\varepsilon}{k}\left(\frac{\hat{P}_k}{\rho\varepsilon}-2f_{\varepsilon}\frac{C_{\varepsilon 2}}{C_{\varepsilon 1}}\right)\end{bmatrix} \qquad \text{(A 15)}$$

$$\hat{P}_k = -\frac{2}{3}\mu_t\left(\frac{\partial u_k}{\partial x_k}\right)\left(\frac{\partial u_k}{\partial x_k}\right)+\mathrm{MIN}\left(0,-\frac{2}{3}\mu_t\bar{\rho}k\delta_{ij}\left[\frac{\partial u_i}{\partial x_j}\right]\right) \qquad \text{(A 16)}$$

Appendix B

Thermodynamic Database Information

The database used by the GUI is an assembly of gaseous and liquid properties for several fuels and oxidizers. Most of the thermodynamic and transport properties were fourth order polynomial curve fits to those provided by McBride et.al.[16] The original data from this report are split into two temperature ranges. These curves have been combined into a single fit for a temperature range 300K to 3000K. The diffusion constants used in the Chapman-Enskog formulation for species diffusion were taken from Svehla.[38] The liquid properties came from Reid.[17] An example of a database entry is shown in Table 15 and described in Table 16 (Superscripts are used to relate Table 15 to the appropriate descriptions in Table 16).

TABLE 15. Sample GUI Database Entry

2^{A}	$CH4O^{B}$	Methanol^{C}	32.0420^{D}			
1^{E}	1^{F}					
		$7.85381449E\text{-}01^{G}$	1.68622442E-02	-9.24980079E-06	2.62973642E-09	-3.03388054E-13
		$-2.23317873E\text{-}06^{H}$	4.24359048E-08	-1.09136602E-11	2.16723754E-15	-1.79831037E-19
		$-1.79325923E\text{-}02^{I}$	9.74961128E-05	6.49563168E-08	-3.10853977E-11	4.26625171E-15
		3.626^{J}	481.8^{K}	$-2.50864348E{+}04^{L}$	$1.97745991E{+}01^{M}$	
		$79.5e5^{N}$	513.0^{O}	337.71^{P}	175.0^{Q}	
		$1178.6e3^{R}$	$2.49e3^{S}$	0.2^{T}	791.5^{U}	
		-7.28936^{V}	1.53679	-3.08367	-1.02456	

TABLE 16. Description of GUI Database Entries

Entry Name	Entry Description
^{A}Entry Number	Database book keeping
^{B}Chemical Name	The chemical name of the species in upper case letters.The GUI is case sensitive. However, a specie such as iso-butane would be iC3H8
^{C}Common Name	A more general specie name
^{D}Molecular Weight	The species molecular weight
^{E}Gas Phase Flag	Indicates to the GUI that a gas phase exists in the database
^{F}Liquid Phase Flag	Indicates to the GUI that a liquid phase exists in the database
^{G}Specific Heat Coefficients	The coefficients used in the fourth order polynomial fit for specific heat
^{H}Viscosity Coefficients	The coefficients used in the fourth order polynomial fit for viscosity
^{I}Thermal Conductivity Coefficients	The coefficients used in the fourth order polynomial fit for thermal conductivity
$^{J}\sigma$	Collision cross section for diffusion
$^{K}T_{\varepsilon}$	Temperature for collision integral correction
^{L}Enthalpy Constant	Integration constant for enthalpy
^{M}Entropy Constant	Integration constant for entropy
^{N}Critical Pressure	Critical pressure for species
^{O}Critical Temperature	Critical temperature for species
^{P}Boiling Temperature	Boiling temperature for species
^{Q}Melting Temperature	Melting temperature for species
^{R}Latent Heat	Latent heat of vaporization at the spray temperature (293K)
^{S}Liquid Specific Heat	Specific heat of the liquid at the spray temperature (293K)
^{T}Liquid Thermal Conductivity	Thermal conductivity of the liquid at the spray temperature (293K)
^{U}Liquid Density	Density of the liquid at the spray temperature (293K)
^{V}Vapor Pressure Coefficients	Coefficients for the polynomial fit to fuel vapor pressure.

ALLSPD3D V1.0 -I/O GUIDE

BC: Boundary Condition
IC: Initial Condition
EQ: Equations
(): Dimensional Units or Index Count

allspd.h - Parameter File

Variable	Description
I_MAX , J_MAX, K_MAX	Maximum Grid Cells in x-y-z direction
IJK_TOTAL	Total Number of Cells
NUM_ZONES	Total Number of Zones
NUM_SPECIES	Total Number Species
NUM_REACT	Total Number of Reactions
MAX_OBSTACLES	Total Number of Obstacles
MAX_DILUTION	Total Number of Dilution Locations(>1)
MAX_TRANSPIR	Total Number of Transpiration Locations(>1)
MAX_SPARKS	Total Number of Spark Ignition Location(>1)
MAX_SPRAY_PARTICLES	Maximum Number of Spray Particles(>1)
MAX_SPRAY_INTEGRATION	Maximum Number of Integration steps(>1)
NUM_FLOW_EQNS	=5
NUM_TURB_EQNS	=2 (fixed)
SCHMIDT_NUM	= 0.9
PRANDTL_NUM	= 0.7
TURB_CONST_K	= 1.0
TURB_CONST_EPS	= 1.3
MAXARGS	= 31 (fixed)

allspd.dat - Input File

Input Variable	Comments
iread	= 0 a fresh run, = 1 a restart run
ntmax	Total number of iterations
nplot	increment at which Plot3d files are generated
nsave	increment at which the restart file is generated
iturb, icmu, idisbc,	iturb, = 0 laminar flow, = 1 turbulent flow, icmu, = 0 constant *C*μ(=0.09), = 1 variable *C*μ idisbc, = 0 Dirichlet ε-wall BC, = 1 Neumann ε wall BC.
icomb	icomb = 0 non-reacting flow, = 1 reacting flow
iartv	iartv = 0 no artificial damping, = 1 with artificial damping
vis4,vis2	vis4 = 4th order damping coefficient, vis2 = 2nd order damping coefficient
CFL()	CFL number, cfl(1) =flow EQ, cfl(2) = κ-ε EQ, cfl(3) = species EQ
prefac()	LU scheme pre-conditioning factor, precfac(1) = flow EQ, precfac(2) = for κ-ε EQ, precfac(3) = species EQ
nswitch	CFL ramping function switch. CFL number is incremented by CFL=CFL×factau from **facini** to 1 over **nswitch** iterations.
facini	Initial CFL factor value(~0.1)
bond	Convergence criterion(default=1e-13)
nkecall	Turbulence EQ update increment
nscall0	Species EQ update increment
iuvwt	= -1 only inlet BC specified by **uin,vin,** and **win** and zero IC, = 0 uniform flow BC and IC, = 1 user provides BC and IC using *uvwt.dat*, = 2 specified swirl BC in *uvwt.dat* file.
uin, vin, win	Initial velocities. uin, vin, win = x-y-z components of velocity (m/s)
Tref	Reference temperature (K).
Pref	Reference pressure (*N/m*^{2})
rlref	A reference length scale(m)
iebm	Eddy break-up model switch, = 0 off(default), = -1 on.
iequate	Equilibrium Model switch, = 0 off, = 1 reverse reaction constant calculation, = -1 forward reaction constant calculation
nt1igt	starting point to add Ignition Energy.
nt2igt	end adding Ignition Energy.
noigt	Number of ignition sources, (if = 0, skips to next input of **toff**)
ibkigt i1gt, i2gt,j...,k...	ignition spark position(s) no*th* ignition is located in **ibkigt(no)** block. **i1igt(no), j1igt(no), k1igt(no):** Starting *i,j,k*-index of the ignition spark location. **i2igt(no), j2igt(no), k2igt(no):** Ending i, j, *k*-index of the ignition spark location.
toff	Cut-off temperature at which the ignition is turned off.
sparkuser	Strength of ignition spark (J/(m^3-s)). =0, (default strength used)
nsdata	= 0, default spray data used, =1, the spray data are to be provided by user-generated file, *spray.dat*.
ispray	= 0 Single-phase flow calculation(skip to input of **inplot(i)**), = 1 Two-phase flow calculation.
izospray	grid block location of spray zone
ispray1,ispray2, **jspray1,jspray2,** **kspray1,kspray2**	Spray nozzle location (i,j,k) between **ispray1, ispray2** points in *i* direction, between **jspray1, jspray2** points in *j* direction, and between **kspray1, kspray2** points in *k* direction.
nangles, angle1, angle2	**nangle** = number of angles increments between (**angle1,angle2**), **angle1** =Lower spray angle (say 10 degree) **angle2** = Upper angle (say 45 degree). Angle measured in *x-z* plane only.
size1, size2	= spray particle size range (in terms of μ*m* in radius).
ntspr	Time step to initiate the spray.
ncalspr	Spray update increment
imspr	= 0 (reserved for later version use)
tispr	Liquid spray temperature(K)
flowispry	Liquid spray mass flow rate (kg/s)
vispr	Liquid spray velocity (m/s)
inplot()	Plot file flags(up to 15), =1 density, = 2, 3, 4, u-v-w momentums, = 5 static temperature, = 6 gage pressure, = 7 static pressure, = 8 turbulent *C*μ, = 9 static enthalpy, = 10 total enthalpy, = 11 turbulent kinetic energy, = 12 turbulent dissipation, = 13 turbulent eddy viscosity, = 14 distance to walls, = i+14 species mass fraction
nsr	Number species in IC
name, Ysi(1)	Species name and Initial species mass fraction

Input Variable	Comments
:	
:	
name, Ysl(ns)	Species name and Initial species mass fraction

gridlink.dat - Grid Parameter Input File

(): block location counter

Input Variable	Comments
Iblank	Flag for *grid.dat*, 0 = non-Iblanked *grid.dat*, 1 = Iblanked *grid.dat*
nblk	total number of grid blocks(<NUM_ZONES)
imaxbl(), jmaxbl(), kmaxbl()	Number of grid points in i, j, and k direction, respectively, in nb*th* block(up to nblk blocks req'd).
least(),lwest(), lnorth(),lsouth(), ltop(),lbot(), bcln((),1), ... ,bcln((),6),	Block BC indices nb*th* block.(up to nblk blocks req'd) BC types, ,=1 upstream (inflow) (all six faces), = 2 downstream (outflow) (all six faces), =3 solid wall (all six faces), = 4 block interface (all six faces), = 5 symmetry (only for north, south, top and bottom faces), = 6 periodic boundary (only for north/south, and top/bottom faces), = 7 singularity (only for bottom face, i.e., k=1), = 8 far-field BC (all six faces). bcln(nb,l) =0 either adiabatic-wall or inlet BC condition. =wall temperature (K) for isothermal wall, = ambient pressure (N/m^2) for exit BC. where, l =1 east face, =2 west face, = 3 north face, =4 south face, = 5 top face, = 6 =bottom face.
lee(),lww(),lnn(), lss(),ltt(),lbb()	Block connectivity = 0, no block interface. lee(nb)=m - nb*th* block's east side borders m*th* block, lnn(nb)=n - nb*th* block's north side borders n*th* block, etc ..
intblk	Internal blockage flag, = 0 no internal blockage skip to dilution holes (ndil) input. =1 assign internal blockage(s) boundary conditions
nintblk(), iintb(nn,()),iinte(nn,()), jintb(nn,()),jinte(nn,()), kintb(nn,()),kinte(nn,())	nintblk(nb): number of internal blockages in nb*th* block. (<MAX_OBSTACLES) For each nnth hole in nbth block iintb(nn,nb), jintb(nn,nb), kintb(nn,nb): beginning *i,j,k*-index of nnth blockage iinte(nn,nb), jinte(nn,nb),kinte(nn,nb): ending i,j,*k*-index of nnth blockage
ndil(), isdil((),nn),iedil((),nn), jsdil((),nn),jedil((),nn) ksdil((),nn),kedil((),nn), dilmass((),nn), diltemp((),nn), diltin((),nn), dilcosx((),nn), dilcosy((),nn), dilcosz((),nn), (dilys((),nn,1...ns)	ndil(nb) number of dilution holes in nbth block.(If zero, skip to next block). (< MAX_DILUTION) For each nnth hole in nbth block isdil(nb,nn), jsdil(nb,nn), ksdil(nb,nn): beginning *i,j,k*-index of nnth dilution site iedil(nb,nn), jedil(nb,nn), kedil(nb,nn): ending *i*,j,k-index nnth dilution site diltemp(nb,nn): temperature (*K*) diltin(nb,nn): turbulence intensity dilcosx(nb,nn), dilcosy(nb,nn), dilcosz(nb,nn): *x,y,z*-directional cosine for cooling velocity dilys(nb,nn,1...ns), mass fractions.
ntrn(()), istrn((),nn),ietrn((),nn), jstrn((),nn),jetrn((),nn), kstrn((),nn),ketrn((),nn) trnmass((),nn), trncosx((),nn), trncosy((),nn), trncosz((),nn), (trnys((),nn,1...ns)	ntrn(nb): number of transpiration cooling holes in nb*th* block. (If zero, skip to next block) (<MAX_TRANSPIR) For each nnth hole in nbth block istrn(nb,nn), jstrn(nb,nn),kstrn(nb,nn): beginning *i*,j,k-index ietrn(nb,nn),jetrn(nb,nn),ketrn(nb,nn): ending *i,j,k*-index trnmass(nb,nn): mass flow rate(kg/m^3) trncosx(nb,nn), trncosy(nb,nn),trncosz(nb,nn): *x*,y,z-directional cosine for cooling velocity. trnys(nb,nn,1...ns) mass fractions

prop.dat - Property Input File

specnm()= name of the species input(< 80 char enclosed with in '')

heading: Datafile Comments(Optional, but blank lines required)

Input Variable	Comments
heading	title
hsf	number of species
heading:	title of the property
specnm(nx),wm(nx) ,nx=1,ns	wm(nx) = molecular weight for each species.
heading	title of the property
specnm(nx), cp0(nx),cp1(nx), cp2(nx),cp3(nx), cp4(nx),cp5(nx), cp6(nx)	curve fit coefficients for specific heat.
heading	title of the property
specnm(nx), eu0(nx),eu1(nx), eu2(nx),eu3(nx), eu4(nx)	curve fit coefficients for dynamic viscosity
heading	title of the property
specnm(nx), cd0(nx),cd1(nx), cd2(nx),cd3(nx), cd4(nx)	curve fit coefficients for thermal conductivity
heading	title of the property
specnm(nx), sigm (nx)	species name, effective collision integral factor, Equation(45)
heading	title of the property
specnm(nx), edk(nx)	species name, effective temperature, Equation(45)
heading	title of the property
heading	Title
nre	Number of reactions

Input Variable	Comments
heading	Title, helpful to write the reaction mechanism (blank lines are acceptable)
heading	Heading for Forward reaction rates
nr, ar(nr), ear(nr), tfrr(nr)	Reaction number, Pre-exponent, Activation Temperature[K], Temperature Correction Factor
heading	Heading for Backward reactions
nr, br(nr), ebr(nr), tbrr(nr)	Reaction number, Pre-exponent, Activation Temperature[K], Temperature Correction Factor (note this is forward only reaction!)
heading	Heading, Forward Stoichiometric Coefficients
nr, xkstp(nr,1..ns)	Reaction number, Forward Stoichiometric Coefficients (There should be nspe of them)
heading	Heading, Backward Stoichiometric Coefficients
nr, xkstpp(nr,ns)	Reaction number, Backward Stoichiometric Coefficients (There should be nspe of them)
heading	Heading Title for Correction Mechanism
nr, thirdO(nr) third(nr,1..ns)	Reaction number, Reaction type, Power of the reaction mechanism in forward direction (for the nspe species)
nr,thirdO(nr) thirdb(nr,1..ns)	Reaction number, Reaction type, Power of the reaction mechanism in Reverse direction (for the nspe species)
pcrl,tcrl,tboil tmelt	Liquid phase information **pcrl** - critical pressure (N/m^2) **tcrl** - critical temperature (K) **tboil**- boiling point temperature (K) **tmelt**- melting point temperature (K)
hfg,cppar,rkpar, rholpar	Liquid phase information **hfg**- latent heat of vaporization (J/kg) **cppar**- specific heat (J/kg-K) **rkpar**- conductivity (J/m-s-K) **rholpar** density (kg/m^3)
vp1,vp2,vp3,vp4	Critical pressure curve fit coefficients

Input Variable	Comments
Note:	**Third0(1)** = -1,Global type kinetics **Third0(1)** = 0, Modified kinetics with fractional power **Third0(1)** >0 detailed kinetics with 3-body corrections $K_f = A \exp(E_a/RT)\ T^0\ [ch3oh]^1\ [02]^1$ $K_b = 0.0$ Mechanism of Westbrook and Dryer[37]

grid.dat **input file format**

For non-Iblanked format:

```
read(iunit,*)nblk
read(iunit,*)(imax(n),jmax(n),kmax(n),n=1,nblk)
do n=1,nblk
read(iunit,*)
(((x(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((y(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((z(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n))
end do
```

For Iblanked format:

```
read(iunit,*)nblk
read(iunit,*)(imax(n),jmax(n),kmax(n),n=1,nblk)
do n=1,nblk
read(iunit,*)
(((x(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((y(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((z(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((iblk(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n))
end do
```

uvwt.dat **input file format(Inlet BC input)**

```
do n=1,nblk
ii=imaxbl(n)
jj=jmaxbl(n)
kk=kmaxbl(n)
do k=1,kk
do j=1,jj
do i=1,ii
read(9) pp, u, v, w, t, tke, tep,(ys(nn), nn=1, ns)
end do
```

nblk=number of blocks, **imaxbl(n)**, **jmaxbl(n)**, **kmaxbl(n)**=grid points in i,j,*k* direction in n*th* block, **pp**=gauge pressure (p-pref), **u**, **v**, **w**=x-y-z components of velocity, **t**=static temperature, **tke**=turbulent kinetic energy, **tep**=turbulent dissipation rate, (**ys(nn)**, **nn=1**, **ns**)=mass fraction for each species

uvwt.dat **input file format(swirl input)**

```
do n=1,nblk
read(iunit,*) irmax
read(iunit,*) rmax,y0,z0
read(iunit,*) (yscup(nx),nx=1,ns)
do ii=1,irmax
read(iunit,*) rcup(ii),ucup(ii),vcup(ii),wcup(ii), tcup(ii),tkcup(ii)
end do
end do
```

nblk=number of blocks, **irmax**=Number of points used to define a swirl profile, **rmax**=radius of the swirler cup (m). **y0**, **z0** =y,z coordinates of the swirler cup center (m), (**yscup(nx)**,**nx=1**,**ns**)= Mass fractions of the species, **rcup(ii)**=r/rmax, where r is the radial distance from the center of the swirler cup **ucup(ii)**, **vcup(ii)**, **wcup(ii)**= u-v-w-velocity (m/s), **tcup(ii)**= temperature (K), **tkcup(ii)**= turbulence intensity

***spray.dat* input file format (spray initialization file)**

```
read(iunit,*) npgrp
do np=1,npgrp
read(iunit,*)
xpar0(np,1),xpar0(np,2),xpar0(np,3),
velpar0(np,1),velpar0(np,2),velpar0(np,3),
tspr0(np),radpar0(np),numpar(np)
end do
```

npgrp = number of group of particles for spray, **xpar0(np,1..3)** = x-y-z coordinate of the initial spray location (m), **velpar0(np,1..3)** = x-y-z component of the spray injection velocity (m/s), **tspr0(np)** = liquid temperature (K) for each group of spray, **radpar0(np)** = radius of each drop size (m), **numpar(np)** = number of droplets per spray group (#/group)

output file format(*.PFS, .PFS1,.PFS2 files)

```
read(iunit,*)nblk
read(iunit,*)(imax(n),jmax(n),kmax(n),n=1,nblk)
do n=1,nblk
read(iunit,*)fm0,angle,reno,time
read(iunit,*)
(((q1(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((q2(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((q3(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((q4(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n)),
(((q5(i,j,k),i=1,imax(n)),j=1,jmax(n)),k=1,kmax(n))
end do
```

fm0 = the free stream Mach number, **angle** = flow angle, **reno** = flow Reynolds number based on rlref, **time** = time, **q1 to q5** = are the flow variables defined in the *allspd.dat*

***spray.SPY* output file format**

```
write(iunit,*) npgrp
do np=1,npgrp
write(iunit,*) np,npmark(np)
write(iunit,*)
xplot(npidx,1),xplot(npidx,2),xplot(npidx,3),
rplot(npidx),tplot(npidx),dtplot(npidx),npidx=1,npmark(n
p))
end do
```

npgrp = number of group of particles for spray, **npmark(np)** = the last number of time integration for each group of particle, **xplot(npidx,1..3)** = x-y-z-coordinate of spray trajectory (m), **rplot(npidx)** = radius of the particle along its trajectory, **tplot(npidx)** = droplet surface temperature along its trajectory (K), **dtplot(npidx)** = integration time step along its trajectory (s)

ALLSPD Code Working Units

Length: meter (m)

Time: second (s)

Mass: kilogram (kg)

Temperature: Degrees Kelvin (K)

Energy: Joules(J)

Force: Newton (N)

Velocity: meter/second (m/s)

Pressure: N/m^2

Density: kg/m^3

Specific heat: J/(kg K)

Dynamic viscosity: kg/(m s)

Kinematic viscosity: m^2/s

The dimensional units required for chemical kinetic rate pre-exponents A_f and A_b are in the M-K-S system.

$$[A_f(k)]\text{or}[A_b(k)] = \left[\frac{\text{kmols}^{1-rm_k}\,\text{m}^{3rm_k-3}}{\text{sec}}\right]$$

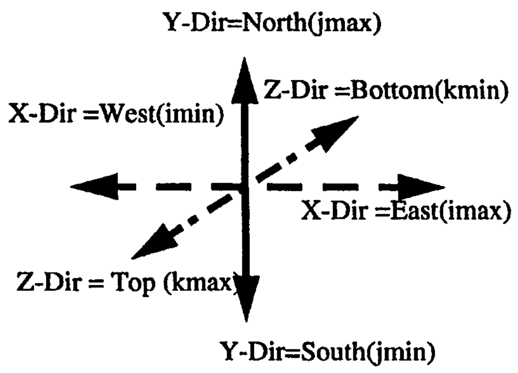
$$rm_k = \sum_{i=1}^{N} v'_{ik} \quad \text{or} \quad \sum_{i=1}^{N} v''_{ik}$$

where the reaction is given by,

$$\sum_{i=1}^{N} v'_{ik}CS_i \leftrightarrows \sum_{i=1}^{N} v''_{ik}CS_i \quad , k = 1,2,3 \ldots \text{Nr}$$

$$\left[\frac{E_f}{R_u}\right]\text{or}\left[\frac{E_b}{R_u}\right] = [K]$$

ALLSPD Grid Orientation



REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (*Leave blank*)

2. REPORT DATE
April 1996

3. REPORT TYPE AND DATES COVERED
Technical Memorandum

4. TITLE AND SUBTITLE
ALLSPD-3D
Version 1.0a

5. FUNDING NUMBERS
WU-505-62-52

6. AUTHOR(S)
K.-H. Chen, B. Duncan, D. Fricker, J. Lee, and A. Quealy

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Lewis Research Center
Cleveland, Ohio 44135-3191
and
Vehicle Propulsion Directorate
U.S. Army Research Laboratory
Cleveland, Ohio 44135-3191

8. PERFORMING ORGANIZATION REPORT NUMBER
E-10197

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546-0001
and
U.S. Army Research Laboratory
Adelphi, Maryland 20783-1145

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA TM-107204
ARL-MR-307

11. SUPPLEMENTARY NOTES
K-H. Chen, Ohio Aerospace Institute, 22800 Cedar Point Road, Brook Park, Ohio 44142; B. Duncan, J. Lee, and A. Quealy, NYMA, Inc., 2001 Aerospace Parkway, Brook Park, Ohio 44142 (work funded by NASA Contract NAS3-27186); D. Fricker, Vehicle Propulsion Directorate, U.S. Army Research Laboratory, NASA Lewis Research Center. Responsible person, D. Fricker, Vehicle Propulsion Directorate, organization code 2650, (216) 433-5960.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified - Unlimited
Subject Categories 01, 05, 34, 61, and 64

This publication is available from the NASA Center for AeroSpace Information, (301) 621-0390.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

This report describes the ALLSPD-3D computer program for the calculation of three-dimensional, chemically reacting flows with sprays. ALLSPD-3D employs a coupled, strongly implicit solution procedure for turbulent spray combustion flows. ALLSPD-3D utilizes generalized curvilinear coordinates with both multi-block and flexible internal blockage capabilities to handle complex geometries. The ALLSPD-3D algorithm, which employs preconditioning and eigenvalue rescaling techniques, is capable of providing efficient solution of flows over a wide range of subsonic Mach numbers. The chemistry treatment in the code is general enough that an arbitrary number of reactions and species can be defined by the users. In addition, for general industrial combustion applications, the code provides both dilution and transpiration cooling capabilities. A stochastic spray model and an efficient method for treating the spray source terms in the gas-phase equations are used to analyze evaporating spray droplets. Although written for three-dimensional flows, the code can be used for two-dimensional and axisymmetric flow computations as well. The code is written in such a way that it can be run on various computer platforms (supercomputers, workstations, and parallel processors) while the GUI (Graphical User Interface) should provide a user-friendly tool for setting up and running the code.

14. SUBJECT TERMS
ALLSPD-3D; User manual; CFD; Gas turbine combustors

15. NUMBER OF PAGES
165

16. PRICE CODE
A08

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT